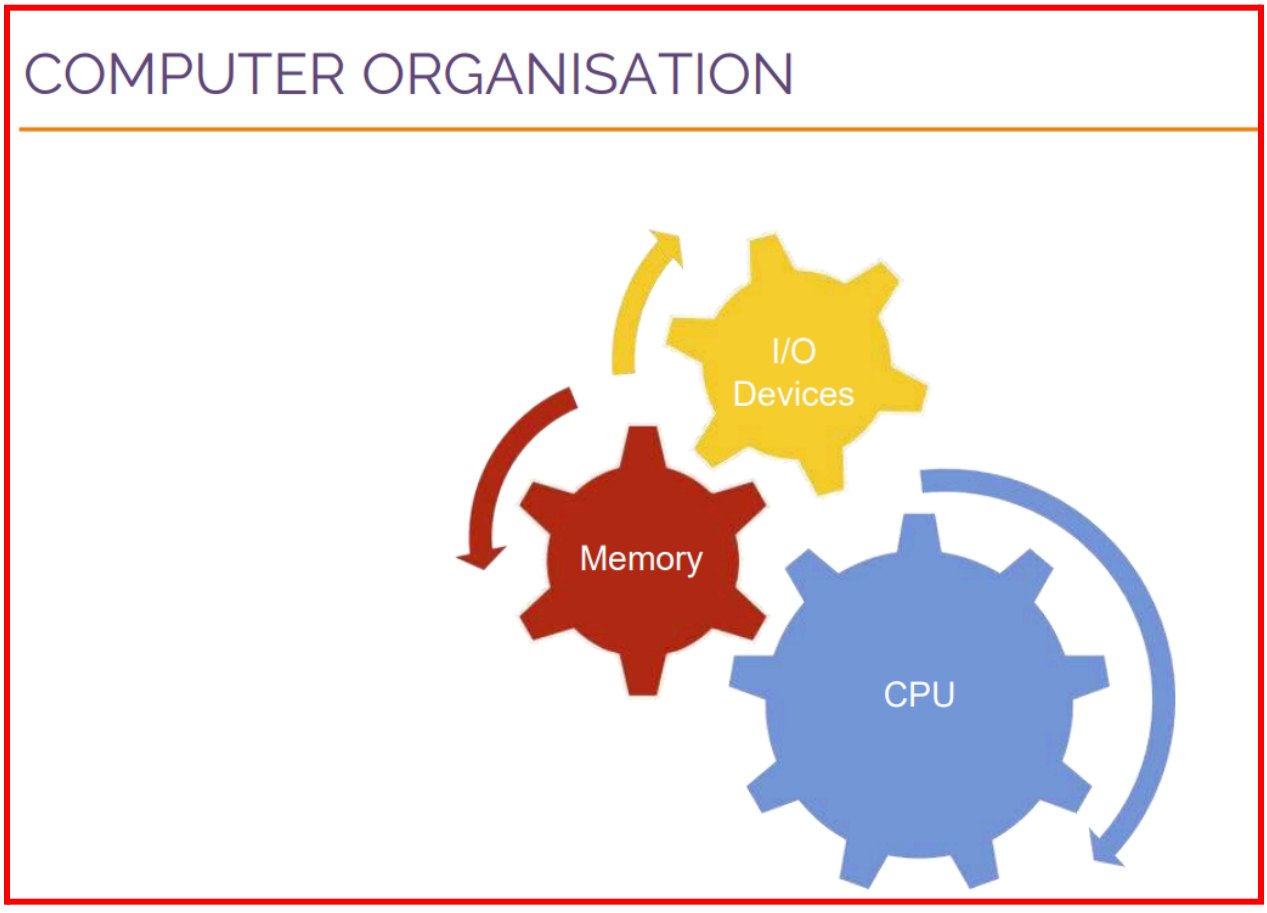
UNIT 4

Understanding Virtualization

Virtualization, Concept of Hypervisor, Types of Hypervisor, Taxonomy of Virtualization, Virtualization and machine reference model, Hardware virtualization techniques, Pros and Cons of Virtualization, Live migration, Technology examples: Xen, KVM, VMware, Microsoft Hyper-V.
Cloud Platforms : AWS, Microsoft Azure, Google Cloud Platform, Architecture, services offered

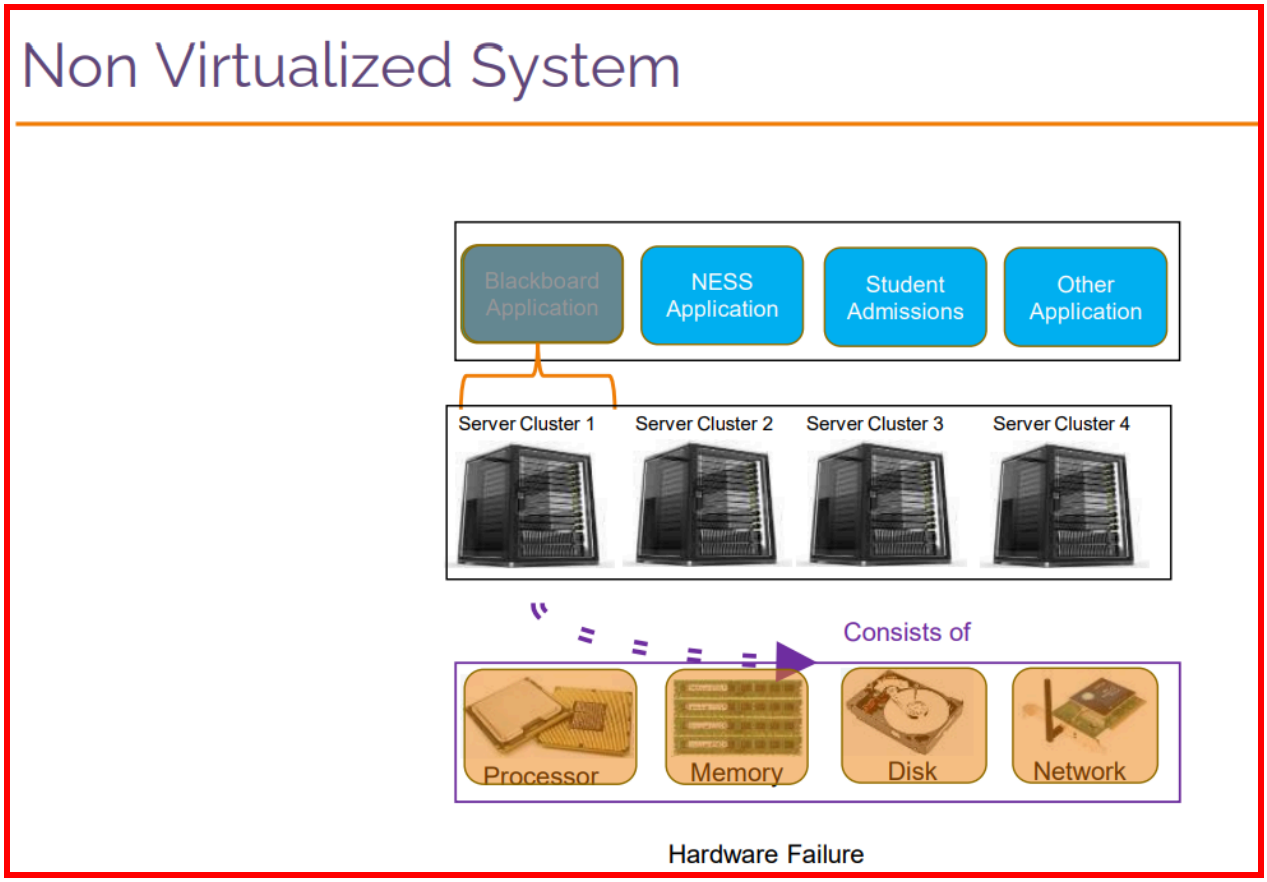


Explanation:

Computer Organisation using gear-like structures to show how different components interact. Here's a simple explanation in points:

1. **CPU (Central Processing Unit)**
    - It is the brain of the computer.
    - It processes instructions and performs calculations.
2. **Memory**
    - It stores data and instructions temporarily.
    - Helps the CPU to access information quickly.
3. **I/O Devices (Input/Output Devices)**
    - These allow users to interact with the computer.
    - Examples: Keyboard, Mouse (Input), Monitor, Printer (Output).
4. **Arrows Indicate Data Flow**
    - Arrows show the movement of data between CPU, Memory, and I/O devices.
    - The CPU processes data from Memory and interacts with I/O devices.

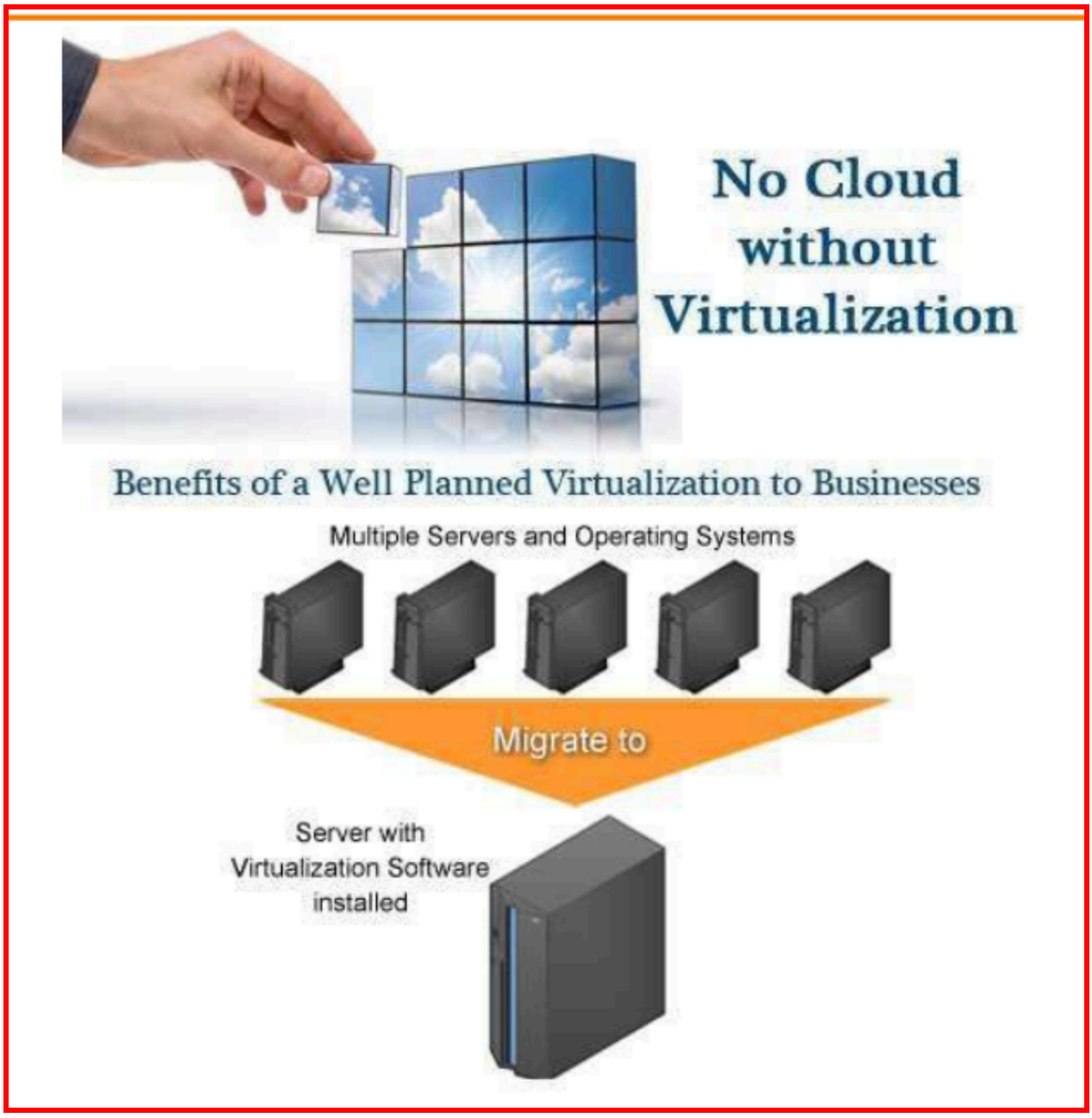
This diagram visually represents how a computer system functions by linking its essential parts together.



1. **What is a Non-Virtualized System?**
    - A system where applications run directly on dedicated physical servers.
    - Each application has its own server or server cluster.
2. **Applications Running**
    - Examples include **Blackboard Application, NESS Application, Student Admissions, and Other Applications**.
    - Each application is assigned to a specific server cluster.
3. **Server Clusters**
    - There are four server clusters shown (**Server Cluster 1, 2, 3, and 4**).
    - Each cluster is a set of physical servers managing assigned applications.
4. **Components of a Server**
    - Each server consists of:
        - **Processor** – Performs computing tasks.
        - **Memory (RAM)** – Stores data for fast access.
        - **Disk (Storage)** – Saves data permanently.
        - **Network** – Connects the system to users and other networks.
5. **Hardware Failure Risk**
    - Since applications depend on physical servers, if hardware fails, the application may stop working.
    - This is a major drawback of non-virtualized systems.

Key Takeaway:

A **Non-Virtualized System** uses separate physical servers for each application, making it dependent on hardware. If a server fails, the application using it can go offline.



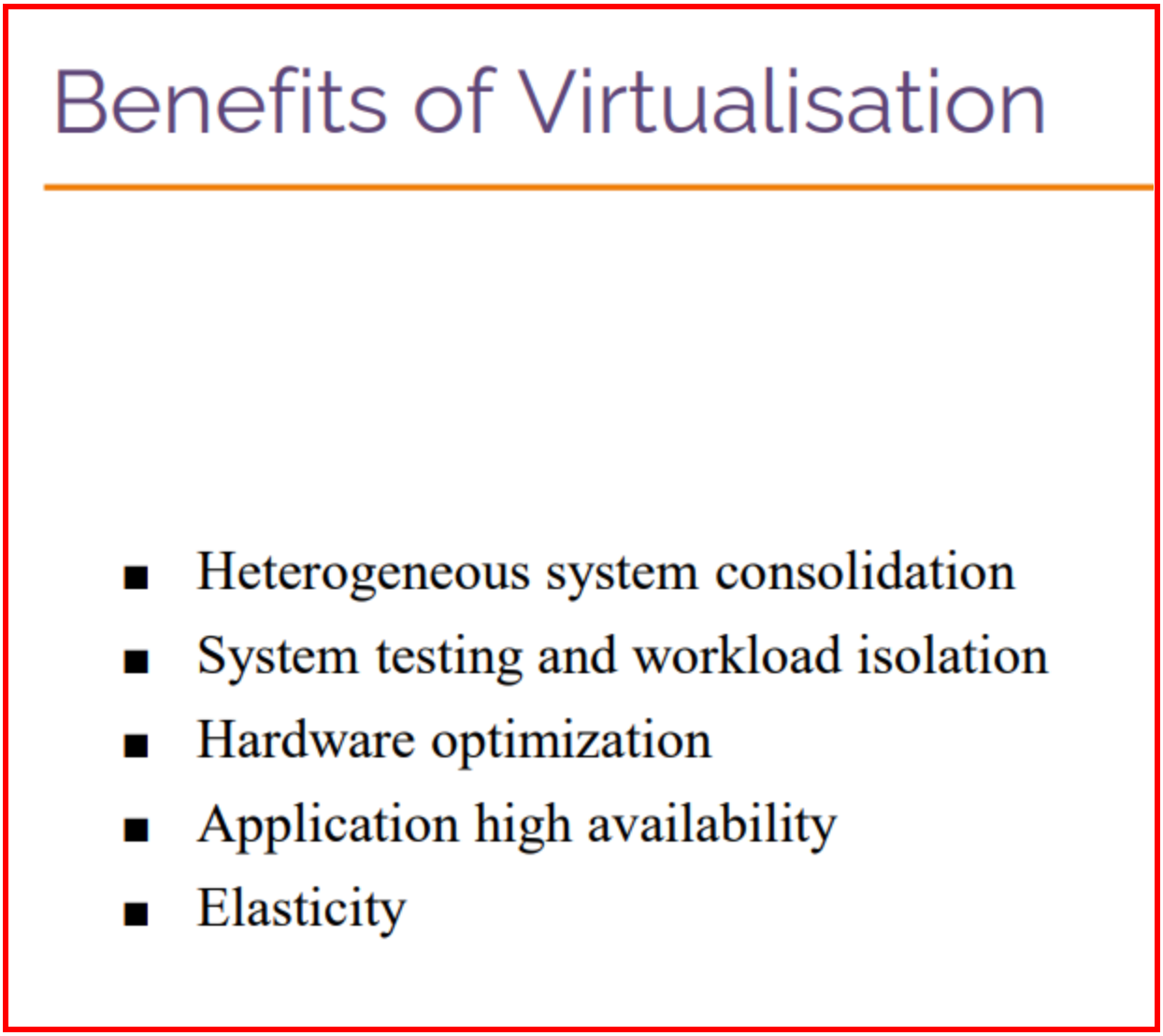
*Explains the **importance of virtualization for cloud computing.***

1. **No Cloud without Virtualization**
    - Cloud computing relies on virtualization to function.
2. **Before Virtualization**
    - Businesses used multiple physical servers, each running a separate operating system.

3. **After Virtualization**
    - All systems are migrated to a **single powerful server** with **virtualization software**.
4. **Benefits**
    - Reduces hardware costs.
    - Increases efficiency and flexibility.
    - Improves resource management.

Key Takeaway:

Virtualization helps combine multiple servers into one, making cloud computing possible.



1. **Heterogeneous System Consolidation**
    - Different types of systems can run together on a single platform.
2. **System Testing and Workload Isolation**
    - Allows safe testing without affecting the main system.
    - Each task runs separately to avoid interference.
3. **Hardware Optimization**
    - Uses hardware efficiently by running multiple virtual machines on one physical server.
4. **Application High Availability**
    - Applications remain available even if hardware fails.

5. **Elasticity**
    - Resources can be adjusted easily based on demand.

Key Takeaway:

Virtualization improves efficiency, flexibility, and reliability in IT systems

Virtualization (Overview)

- Fundamental component of cloud computing, especially in IaaS.
- Allows the creation of secure, customizable, and isolated execution environment, even if untrusted, **NOT** affecting other users' applications.
- **Basic idea:** ability of a computer program (software and hardware) **to emulate** an executing environment separate from the one that hosts such programs.
- For example, Windows OS can run on top of a virtual machine, which itself is running on Linux OS.
- A great opportunity to build elastically scalable systems
- Provision capability with minimum costs.
- Virtualization used to deliver customizable computing environments on demand.

1. **Important for Cloud Computing**
    - Virtualization is a key part of cloud computing, especially in **IaaS (Infrastructure as a Service)**.
2. **Secure and Isolated Environments**
    - It creates safe, customizable, and separate environments without affecting other applications.
3. **Emulation of Systems**
    - Virtualization allows a program to **simulate** another environment.
4. **Example**
    - **Windows OS** can run on a virtual machine inside **Linux OS**.
5. **Scalability**
    - Helps build systems that can easily grow and adapt to demand.
6. **Cost-Effective**

    - Reduces costs by efficiently using resources.
7. **Custom Computing Environments**
    - Provides flexible and on-demand computing setups.

Key Takeaway:

Virtualization makes computing more **efficient, scalable, secure, and cost-effective**.

- Large umbrella of technologies & concepts to provide abstract environment (**virtual hardware: processing elements (PEs), storage, memory, and networking or an operating system**).
- Synonymous with hardware virtualization to deliver IaaS.
- Operating system level, programming language level and application level.
- **Interested phenomena:**
  - Increased performance and computing capacity.
  - Underutilized hardware and software resources.
  - Lack of space.
  - Greening initiatives.
  - Administrative costs.

*That virtual computer exists because of **hardware virtualization**. This whole service is called **IaaS**.*

1. **What is Virtualization?**
    - Uses technology to create **virtual hardware** like processing power, storage, memory, and networking.
2. **Connection to Cloud Computing**
    - Helps in delivering **IaaS (Infrastructure as a Service)** through hardware virtualization.
3. **Different Levels of Virtualization**
    - Can happen at the **operating system, programming language, and application level**.

4. **Key Benefits and Challenges**

    - **Increases performance** and computing power.
    - **Reduces unused hardware** and software resources.
    - Helps in solving **space limitations**.
    - Supports **environment-friendly (green) initiatives**.
    - **Manages administrative costs** efficiently.

Key Takeaway:

Virtualization **optimizes resources, improves performance, and supports cloud computing**.

Virtualization reference model

- Virtual version of hardware, a software environment, storage, or a network.
- Three major components: Guest, Host and, Virtualization layer.
- The Guest represents the system component that interacts with the virtualization layer.
- Host represents original environment, the guest is supposed to be managed.
- Virtualization layer recreates the environment where the guest will operate.
- Most intuitive application is represented by hardware virtualization, the guest is represented by a system image comprising an operating system and installed applications.

1. **What is Virtualization?**

    - It creates a **virtual version** of hardware, software, storage, or a network.
2. **Three Main Components:**

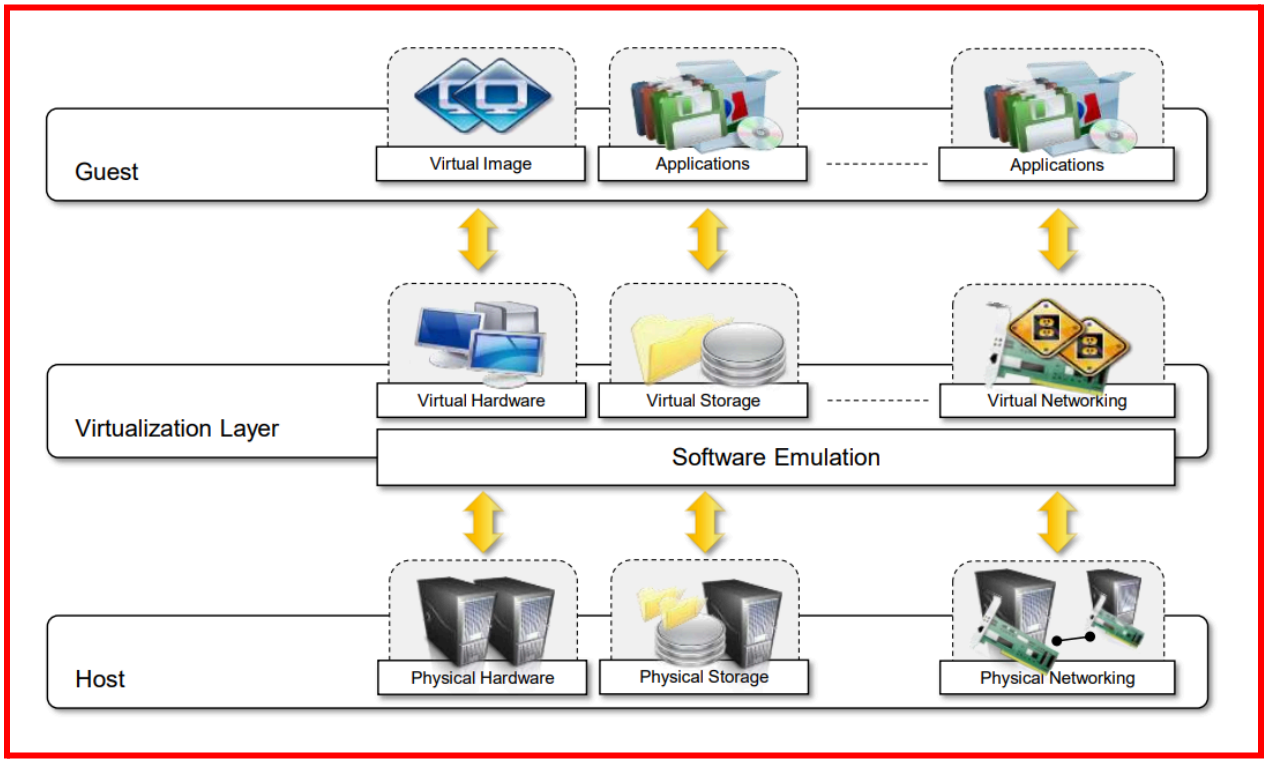
    - **Guest** – The system or application that runs in the virtual environment.
    - **Host** – The actual physical system where the guest runs.
    - **Virtualization Layer** – The software that creates and manages the virtual environment.
3. **How It Works:**

    - The **Guest** interacts with the **Virtualization Layer**.
    - The **Host** is the real system that supports the **Guest**.
    - The **Virtualization Layer** allows the Guest to function as if it is running on real hardware.
4. **Example:**

- A Windows OS can run as a **Guest** inside a Linux **Host** using virtualization software.

Key Takeaway:

Virtualization **allows multiple systems to run on a single machine, making computing more flexible and efficient**.



Explanation: For virtual model(the above diagram)

1. **Three Main Parts:**
    - **Guest** – The virtual system running applications.
    - **Virtualization Layer** – Converts physical resources into virtual resources.
    - **Host** – The actual physical hardware that supports virtualization.
2. **Guest (Top Layer):**
    - Contains **Virtual Images** (like an operating system).
    - Runs **Applications** inside the virtual environment.

3. **Virtualization Layer (Middle Layer):**
    - Creates **Virtual Hardware** (mimics real computers).
    - Provides **Virtual Storage** (like a virtual hard drive).
    - Manages **Virtual Networking** (simulates internet and connections).
    - Uses **Software Emulation** to make virtual systems act like real ones.
4. **Host (Bottom Layer):**
    - Consists of **Physical Hardware** (actual machines).
    - Provides **Physical Storage** (hard drives, SSDs).
    - Connects to **Physical Networking** (real internet and network).

Key Takeaway:

The **Virtualization Layer** helps convert real hardware into **virtual machines**, allowing multiple systems to run on a single physical machine efficiently.

- Guests are installed on top of virtual hardware, controlled and managed by the virtualization layer (virtual machine manager ((VMM)).
- Host is instead represented by the physical hardware, and in some cases the operating system that defines the environment where the VMM is running.
- **Virtual storage**: guest might be client applications interact with the virtual storage management software deployed on top of the real storage system.
- Virtual networking: guest interacts with a virtual network (VPN) managed by specific software (VPN client) using the physical network on the node.
- VPNs are useful for creating the illusion of being within a different physical network and thus accessing the resources in it, which would otherwise not be available.

Explanation for above 👍

1. **Guest Systems**
    - Guests are virtual machines running on virtual hardware.
    - They are managed by the **Virtual Machine Manager (VMM)**.
2. **Host System**
    - The host is the actual **physical hardware**.
    - Sometimes, the host also includes the **operating system** that runs the virtualization software.

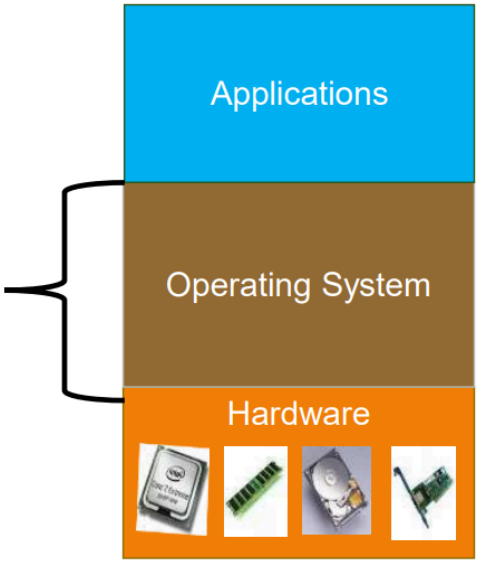
3. **Virtual Storage**
    - Virtual machines use **virtual storage** instead of physical hard drives.
    - The guest system interacts with **storage management software**, which manages data on actual storage devices.
4. **Virtual Networking**
    - Virtual machines can use a **Virtual Private Network (VPN)**.
    - A VPN allows the guest to connect to networks securely, using the physical network of the host.
5. **VPN Benefits**
    - A VPN makes it look like the system is part of another network.
    - This allows access to resources that would normally be unavailable.

Key Takeaway:

Virtualization allows multiple **virtual machines** to share hardware, storage, and networks efficiently, making computing more flexible and secure.

Virtualisation in conventional software infrastructure

Physical Resource	Virtual Resource
CPU	Process
Memory	Virtual Memory
Disk	Files
Network Card	TCP/UDP Sockets
Screen	Windows



Explanation of Virtualization in Software Infrastructure(the above diagram)

1. **What is Virtualization?**
    - It converts **physical hardware** into **virtual resources** to improve efficiency.
2. **Mapping of Physical to Virtual Resources:**
    - **CPU → Process** (Virtualization allows multiple processes to run on a single CPU).
    - **Memory → Virtual Memory** (Extends available memory using storage).
    - **Disk → Files** (Data stored as files instead of raw disk space).
    - **Network Card → TCP/UDP Sockets** (Virtual networks replace physical network connections).
    - **Screen → Windows** (Graphical windows replace direct screen access).
3. **Software Layers in Virtualization:**
    - **Hardware Layer** – Includes physical components like CPU, RAM, and disk.
    - **Operating System Layer** – Manages hardware and runs applications.
    - **Application Layer** – Users interact with software programs running on the system.

Key Takeaway:

Virtualization allows multiple applications to **share hardware efficiently**, making systems more **flexible and resource-efficient**.

What is Virtualization by OS?

Virtualization by the OS means it creates a layer of **abstraction** between the physical hardware and the software (applications), so that each application thinks it's working on its **own private resources**, even though they are shared.

Now let's go **resource by resource** and see **how the OS virtualizes** them:

1. CPU → Process

- **Real CPU:** One or a few cores.
- **Virtualization by OS:**
    - OS uses **time-sharing** (scheduling) to give each process a slice of CPU time.

- Each process "feels" like it has the CPU to itself.
- This is managed by the **scheduler**.

Result: Multiple programs run as if they each have their own processor.

2. Memory → Virtual Memory

- **Real Memory (RAM):** Limited, physical.
- **Virtualization by OS:**
  - OS assigns **virtual addresses** to each program.
  - Uses a **memory management unit (MMU)** to map virtual addresses to physical addresses.
  - Allows **isolation**: One program can't access another's memory.
  - Uses **paging and swapping** for memory overflow.

Result: Each program thinks it has its own private, continuous block of memory.

3. Disk → Files

- **Real Disk:** Raw storage blocks.
- **Virtualization by OS:**
  - OS uses a **file system** (like NTFS, ext4) to manage and organize files.
  - Provides a high-level interface: files, directories.
  - Handles access permissions, metadata, file paths, etc.

Result: Programs see structured files, not raw blocks.

4. Network Card → Sockets

- **Real NIC (Network Interface Card):** A single physical interface.
- **Virtualization by OS:**
    - OS provides **sockets (TCP/UDP)** to apps.
    - Multiple apps can communicate over the network using these sockets.
    - OS handles routing, ports, and protocol details.

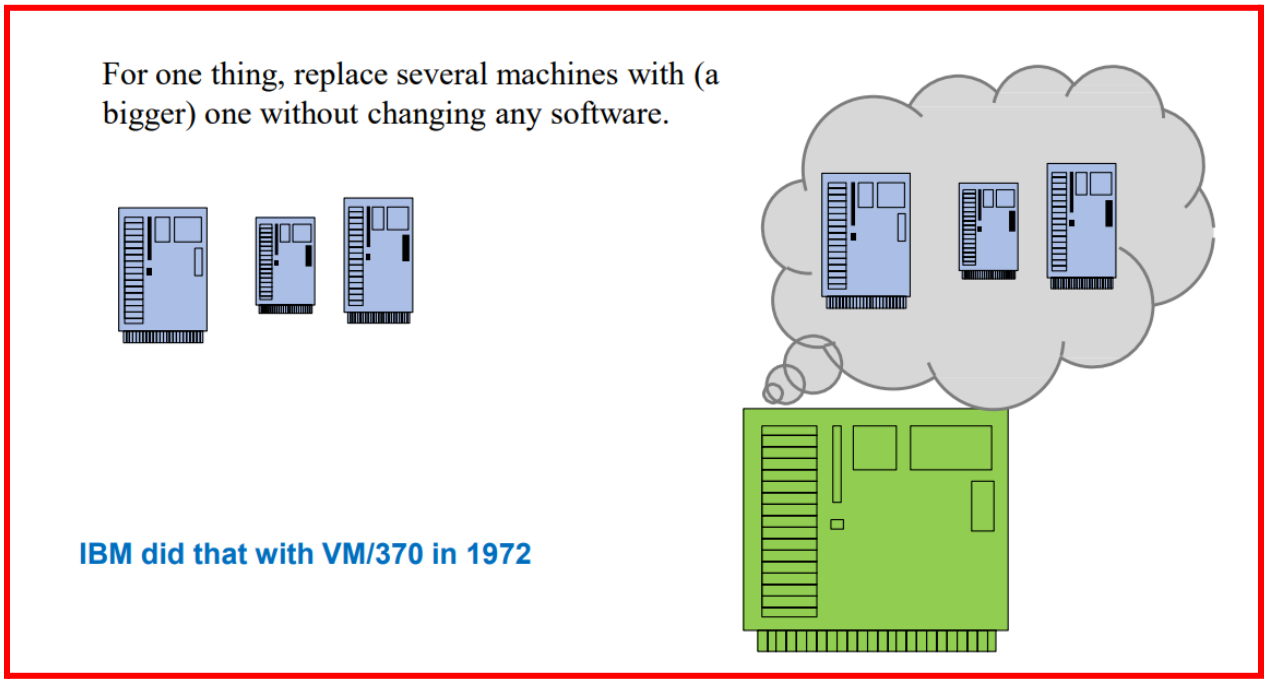
Result: Multiple apps can independently use the same network card.

5. Screen → Windows

- **Real Screen:** One physical display.
- **Virtualization by OS:**
    - OS gives each app its own **window** on the screen.
    - Each window is like a **virtual screen**.
    - Managed by the **windowing system** (like X Window, Windows GUI).

Result: Apps think they control their own display.

What do we want to achieve with Virtualization in clouds?



Explanation of **Virtualization in Cloud Computing:**

1. **Purpose of Virtualization in Clouds**
    - Replace multiple smaller machines with **one powerful machine**.
    - No need to change existing software while doing so.
2. **How It Works**
    - Instead of using **many physical machines**, virtualization allows running multiple virtual machines on a **single large server**.
3. **Efficiency**
    - Saves **hardware costs** by using fewer machines.
    - Increases **performance** and **resource utilization**.
4. **IBM's Contribution**
    - **IBM introduced VM/370 in 1972**, one of the first virtualization systems.
    - It allowed multiple virtual machines to run on a **single mainframe computer**.

Key Takeaway:

Virtualization in cloud computing **reduces the need for multiple physical machines, making systems more cost-effective, efficient, and scalable**.

WHY DO THAT?

- Save the cost (space, energy, personnel) of running several machines in place of one—a **green** aspect, too!
- Use the (otherwise wasted) CPU power
- *Clone* servers (for example, for debugging) at low cost
- *Migrate* a machine (for example, when the load increases) at low cost
- *Isolate* an appliance—a server for a specific purpose (such as security)—without buying new hardware

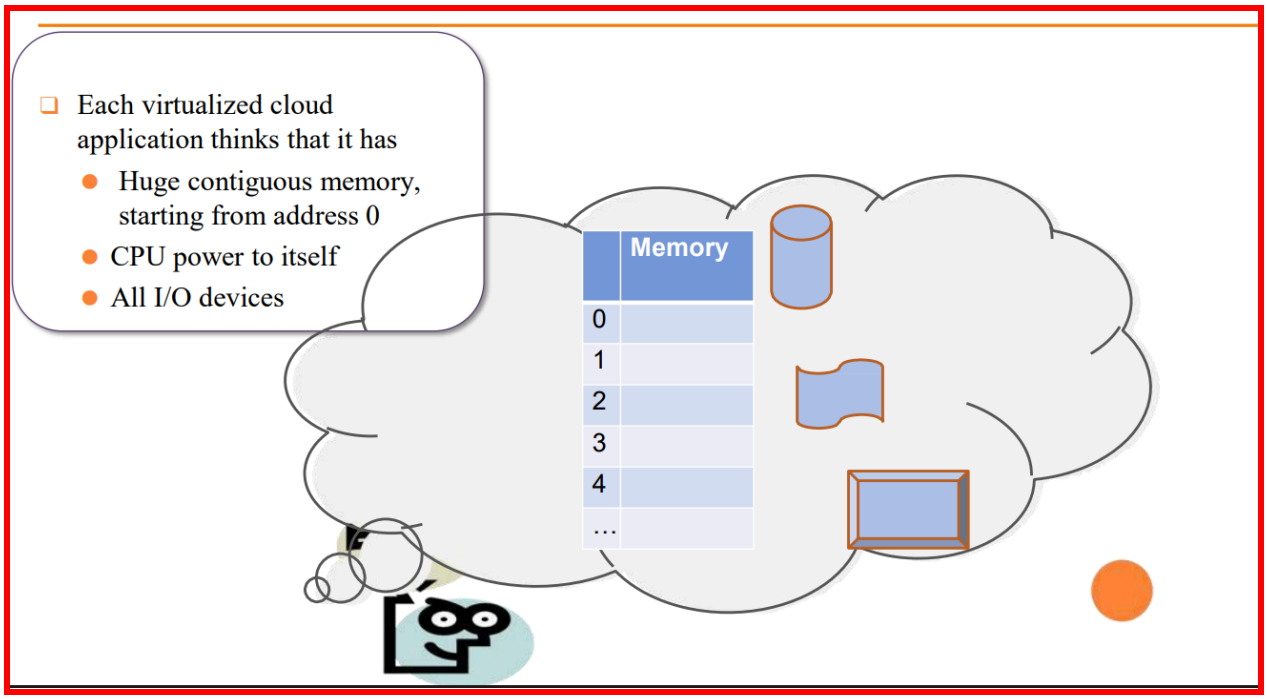
Why Use Virtualization? (Simple Explanation)

1. **Save Costs**
    - Reduces expenses on **space, energy, and staff** by using fewer machines.
    - Helps in **environmental conservation (green technology)**.
2. **Utilize CPU Power Efficiently**
    - Prevents **wasting unused computing power**.
    - Maximizes performance of available hardware.
3. **Clone Servers Easily**
    - Can create **copies of servers** for testing and debugging at a **low cost**.
4. **Migrate Systems Smoothly**
    - Move a system to another server when **workload increases** without downtime.
5. **Isolate Applications for Security**
    - Assign specific servers for tasks like **security** without buying new hardware.

Key Takeaway:

Virtualization **saves money, improves efficiency, and enhances security**, making IT infrastructure more flexible and cost-effective.

VIRTUALIZATION IN MODERN OPERATING SYSTEM



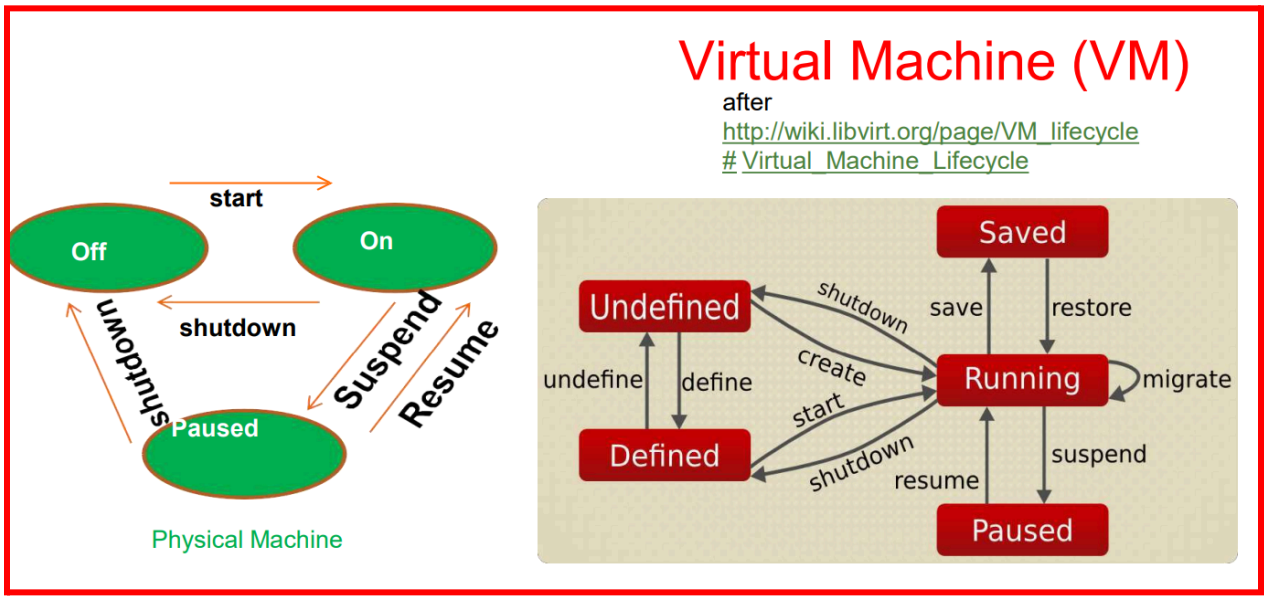
Virtualization in Modern Operating Systems:

1. **What Happens in Virtualization?**
    - Each virtualized cloud application **believes** it has its own system resources.
2. **Virtualized Applications Think They Have:**
    - **Huge continuous memory** (starting from address 0).
    - **Dedicated CPU power** (as if the processor is only for them).
    - **Full control over Input/Output (I/O) devices** (such as storage, network, and peripherals).
3. **Reality of Virtualization:**
    - In reality, **multiple virtualized applications** share the same physical hardware.
    - The **operating system manages** these virtual environments.

Key Takeaway:

Virtualization makes each application think it has its own **CPU, memory, and devices**, while actually **sharing resources** efficiently.

CASE STUDY: LIFECYCLE



Explanation for above :**Virtual Machine (VM) Lifecycle**

1. **Physical Machine Lifecycle**

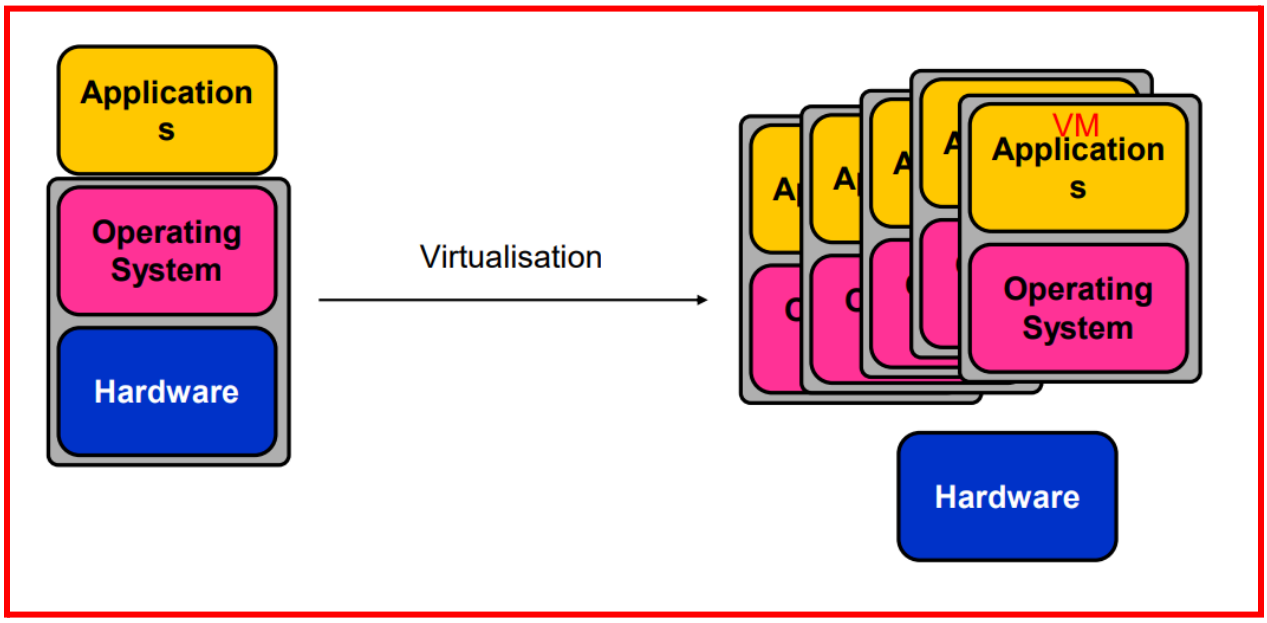
    - **Off** → The machine is powered off.
    - **On** → The machine is running.
    - **Paused** → The machine is temporarily stopped but can resume quickly.
    - **Suspend/Resume** → Saves the current state and resumes later.
    - **Shutdown** → Turns off the machine completely.
2. **Virtual Machine (VM) Lifecycle**

    - **Undefined** → The VM does not exist yet.
    - **Defined** → The VM is created but not running.
    - **Running** → The VM is active and working.
    - **Paused** → The VM stops temporarily but can resume quickly.
    - **Saved** → The VM's current state is stored to be restored later.
    - **Shutdown** → The VM is powered off.
    - **Migrate** → The VM is moved to another system.

Key Takeaway:

A **virtual machine can be created, paused, saved, resumed, or migrated**, just like a physical machine but with more flexibility.

Virtualization: a Technique for Multiple Applications/OS



Virtualization: A Technique for Multiple Applications/OS ::

1. What is Virtualization?

- Virtualization is a technology that allows **multiple operating systems (OS) and applications** to run on a **single physical machine**.
- It creates **Virtual Machines (VMs)** that act like separate computers but share the same hardware.

2. Understanding the Diagram

Before Virtualization (Left Side of Diagram)

- A **single operating system (OS)** runs directly on the hardware.
- This OS supports **only one set of applications** at a time.
- The hardware is dedicated to one OS, **limiting flexibility and resource use**.

After Virtualization (Right Side of Diagram)

- Multiple **Virtual Machines (VMs)** run on the same hardware.
- Each **VM has its own OS** and applications.
- The hardware is shared efficiently, improving **scalability and flexibility**.

3. Key Benefits of Virtualization

✔ Better Resource Utilization

- Instead of dedicating one hardware to one OS, virtualization **allows multiple OS to share the same physical machine**, making better use of computing power.

✔ Cost Savings

- Reduces the need to buy multiple physical servers, **saving hardware costs**.
- Lowers electricity and maintenance expenses.

✔ Isolation & Security

- Each **VM is independent**, so if one crashes, it does not affect others.
- Useful for testing applications in a **secure and isolated** environment.

✔ Scalability & Flexibility

- Easily **create, clone, and delete VMs** based on business needs.
- Allows running different **operating systems (Windows, Linux, macOS) on the same hardware**.

✔ Easy Migration & Backup

- VMs can be **migrated** from one physical server to another without downtime.
- Snapshots and backups can be taken **quickly and restored easily**.

4. How Virtualization Works?

- A **hypervisor (virtualization software)**, such as **VMware, VirtualBox, or Hyper-V**, creates and manages **multiple virtual machines (VMs)**.
- Each VM behaves like a **separate physical computer**, but all share the **same physical hardware**.

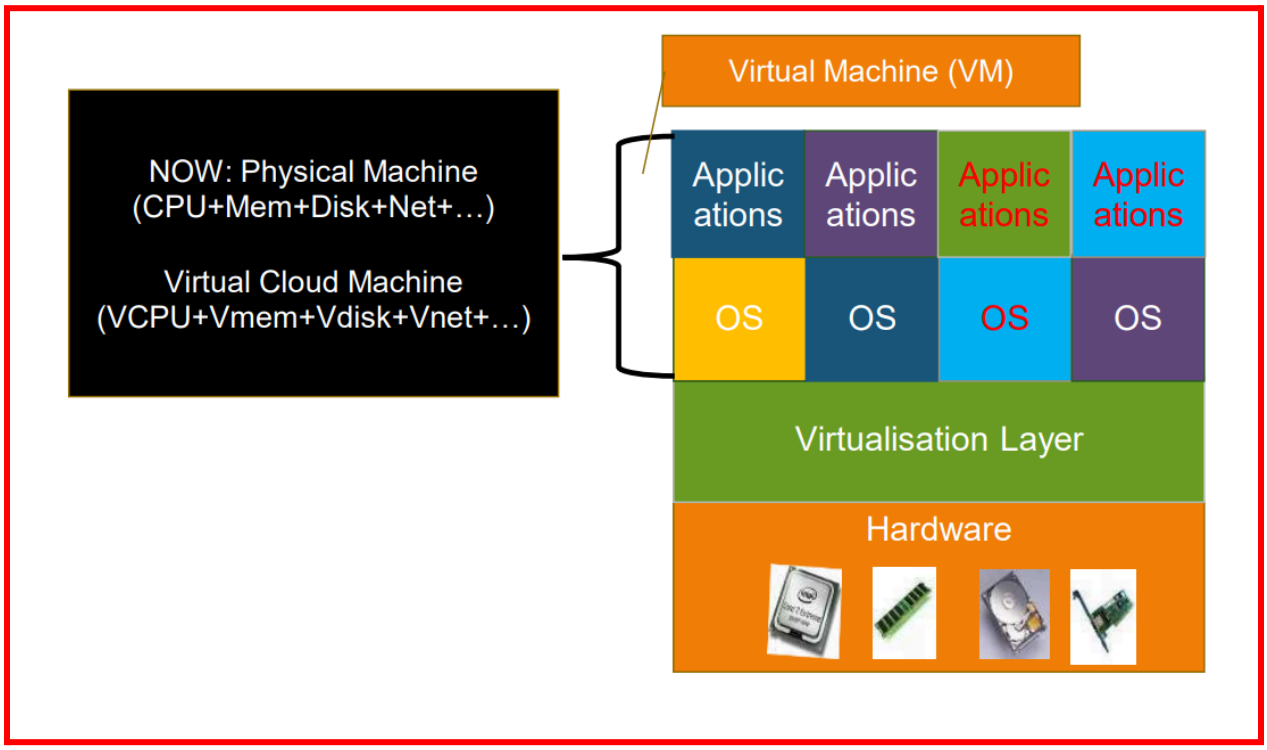
5. Real-Life Use Cases

- **Cloud Computing** – Companies like Amazon AWS, Google Cloud, and Microsoft Azure use virtualization to run multiple customer applications on shared hardware.
- **Software Development & Testing** – Developers test applications on different OS without needing multiple devices.
- **IT Infrastructure Optimization** – Businesses reduce the number of physical servers by running multiple VMs on a single machine.

🔹 Key Takeaway:

Virtualization **maximizes efficiency, reduces costs, improves flexibility, and enhances security** by allowing multiple operating systems and applications to run on the same hardware.

Virtualized Infrastructure in Clouds



Virtualized Infrastructure in Clouds

1. What is Virtualization in Cloud Computing?

- Virtualization allows a **single physical machine** to run multiple **virtual machines (VMs)**.
- This makes cloud computing **more scalable, flexible, and cost-effective**.

2. Key Components Explained

A. Physical Machine (Before Virtualization) – Left Side of Diagram

- A traditional system has:
    - **CPU (Processor)** – Runs computations.
    - **Memory (RAM)** – Temporary storage for quick access.
    - **Disk (Storage)** – Permanent data storage.
    - **Network (Internet & Communication)** – Connects the system to networks.

B. Virtual Cloud Machine (After Virtualization) – Right Side of Diagram

- Instead of using physical hardware directly, virtualization creates **virtual resources**:
    - **vCPU** – A virtual processor instead of a physical one.
    - **vMemory** – Virtual memory allocated to each VM.
    - **vDisk** – Virtual storage for each VM.
    - **vNetwork** – Virtual network connections.

3. Layers of Virtualized Cloud Infrastructure

✔ Hardware Layer (Bottom)

- The **actual physical components** (CPU, RAM, Hard Drive, Network).
- These resources are shared among multiple virtual machines.

✔ Virtualization Layer (Middle)

- Creates and manages **Virtual Machines (VMs)**.
- Ensures that **each VM gets its own allocated resources**.
- Examples of virtualization software: **VMware, Hyper-V, KVM, VirtualBox**.

✔ Operating System Layer (Above Virtualization Layer)

- Each VM has its **own operating system**.
- Different OS types (Windows, Linux, macOS) can run **simultaneously** on the same hardware.

✔ Application Layer (Top Layer)

- Each VM runs **separate applications**.
- No interference between applications running on different VMs.

4. Benefits of Virtualized Cloud Infrastructure

✔ Efficient Resource Utilization

- Multiple virtual machines **share the same hardware**, reducing hardware waste.

✔ Cost Savings

- Fewer physical machines mean **lower costs** for hardware, power, and maintenance.

✔ Scalability & Flexibility

- Can **quickly create or delete VMs** based on demand.
- Perfect for **cloud computing environments** like AWS, Google Cloud, and Azure.

✔ Improved Security & Isolation

- If one VM crashes, **it does not affect other VMs**.
- Each VM can have **different security settings and access controls**.

✔ Disaster Recovery & Migration

- Virtual Machines can be **backed up, restored, and migrated** easily without downtime.

5. Real-World Use Cases

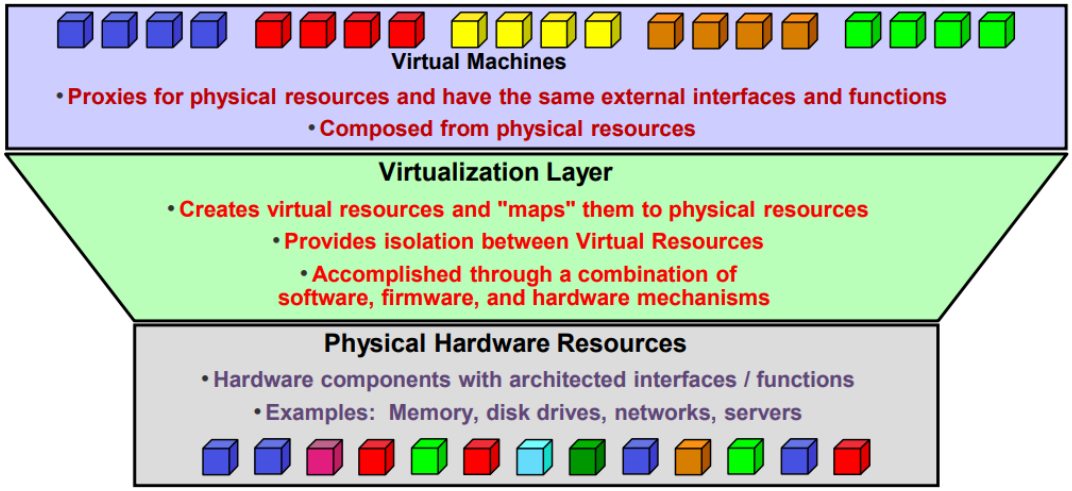
- **Cloud Computing** – AWS, Azure, and Google Cloud use virtualization to run multiple users' workloads on shared hardware.
- **Software Testing** – Developers test applications on different operating systems without needing separate machines.
- **Data Centers** – Companies run multiple VMs on fewer physical servers, **saving space and energy**.

🔹 Key Takeaway:

Virtualization in cloud computing **allows multiple operating systems and applications to run on shared hardware efficiently**, reducing costs, improving security, and making systems scalable.

Virtualized Infrastructure in Clouds

Server Virtualization – The division of a physical system's resources (e.g. processors, memory, I/O, and storage), where each such set of resources operates independently with its own System Image instance and applications



1. **What is Server Virtualization?**

- Server Virtualization **divides a physical system's resources** (CPU, memory, storage, I/O, and network).
- Each virtual machine (VM) operates **independently with its own operating system and applications**.
- This allows **multiple systems** to share the same physical hardware efficiently.

2. Understanding the Three Layers in Virtualization

A. Physical Hardware Resources (Bottom Layer)

- The **real physical components** of a system:
  - **Processors (CPU)** – Runs all computing tasks.
  - **Memory (RAM)** – Stores temporary data for fast access.
  - **Disk Storage (HDD/SSD)** – Stores files and OS permanently.
  - **Networking (NIC, Routers, etc.)** – Enables communication between systems.

B. Virtualization Layer (Middle Layer)

- The **core of virtualization**, which acts as a bridge between physical and virtual resources.
- **Key Functions:**

  ✔ **Creates Virtual Resources** – Converts physical CPU, memory, and disk into virtual

ones.

✔ **Maps Virtual Machines to Physical Hardware** – Allocates a portion of CPU, RAM, and storage to each VM.

✔ **Ensures Isolation** – Each VM runs separately, **without interfering** with others.

✔ **Uses Software, Firmware, and Hardware Mechanisms** – Managed by **hypervisors** like **VMware, Hyper-V, KVM, or VirtualBox**.

C. Virtual Machines (Top Layer)

- Virtual Machines (VMs) **run on top of the Virtualization Layer** and act like real computers.
- Each VM has:
    - **Its own Operating System (OS)** (Windows, Linux, macOS, etc.).
    - **Its own Applications** (Software installed within the VM).
    - **Acts as a Proxy for Physical Resources** – Uses virtualized CPU, memory, and storage but behaves like an independent machine.

3. Key Benefits of Server Virtualization

✔ Better Resource Utilization

- Instead of dedicating a physical machine to one task, multiple VMs can share the same **CPU, RAM, and storage**.

✔ Cost Savings

- Reduces the need for purchasing multiple physical servers.
- Saves on **electricity, maintenance, and cooling** costs.

✔ Improved Security & Isolation

- If **one VM crashes**, it **does not affect** other VMs.
- Each VM can have **different security policies**.

✔ Scalability & Flexibility

- New VMs can be **created, cloned, or deleted easily**.
- Ideal for **cloud computing environments** like AWS, Azure, and Google Cloud.

✔ Disaster Recovery & Backup

- VMs can be **backed up, restored, and migrated** between different physical servers **without downtime**.

4. Real-World Use Cases

- **Cloud Computing** – AWS, Azure, and Google Cloud use virtualization to host multiple users on shared infrastructure.
- **Software Testing & Development** – Developers can test applications on multiple OS without buying new hardware.
- **Enterprise IT Infrastructure** – Companies consolidate multiple workloads onto fewer physical machines, saving costs.
- **High Availability Systems** – Virtual machines can be quickly **moved to another physical machine** if hardware fails.

🔹 Key Takeaway:

Server Virtualization **divides physical resources into multiple independent virtual machines (VMs), improving efficiency, flexibility, and cost-effectiveness in cloud computing**.

Characteristics of virtualized environments

Increased security :

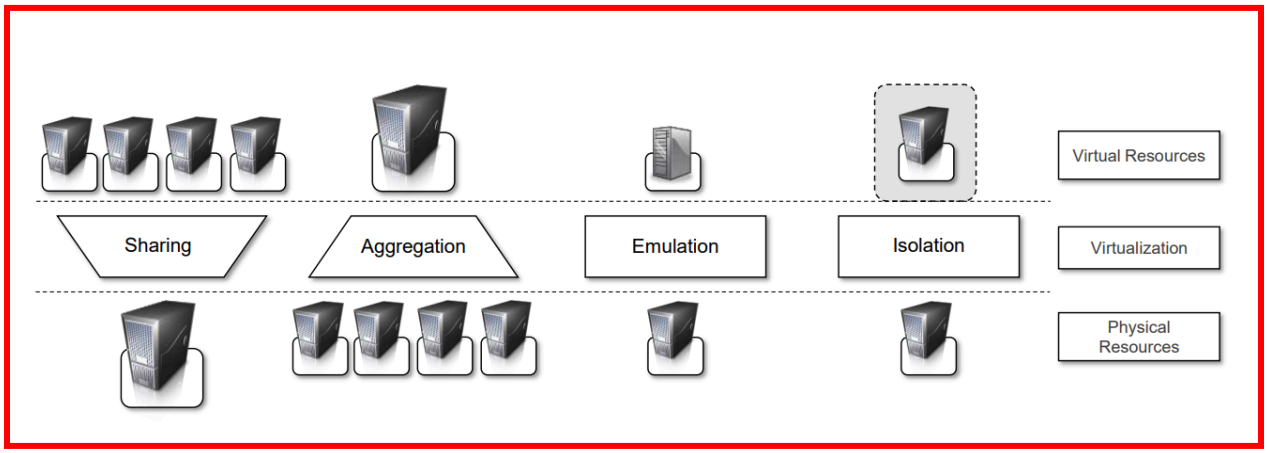
- For guest, a completely transparent and controlled execution environment.
- An emulated environment in which the guest is executed.
- Guest operations performed on VM, It translates and applies them to the host.
- VMM controls and filters the guest activity, thus preventing some harmful operations from being performed, Host is hidden or protected from the guest.
- Sensitive information naturally hidden without installing complex security policies.
- Increased security is a requirement when dealing with untrusted code. For example, applets downloaded from the Internet run in a sandboxed version o f the Java Virtual Machine(JVM), which provides them with limited access to the hosting operating system resources.

Increased security :

- Hardware virtualization solutions (VMware Desktop, Virtual Box, and Parallels) create a virtual computer with customized virtual hardware on top of which a new operating system can be installed.
- By default, the file system exposed by the virtual computer is completely separated from the one of the host machine.

Managed execution :

- Sharing: sharing in virtualized data centers, to reduce the number of active servers and limit power consumption
- Aggregation: A group of separate hosts tied together and represented to guests as a single virtual host, implemented in middleware for distributed computing.
- Emulation Controlling and tuning the environment that is exposed to guests.
- Isolation Allows multiple guests to run on the same host without interfering with each other, Also, Separation between the host and the guest.



Characteristics of virtualized environments

Portability:

- **Hardware virtualization solution:** guest packaged into a virtual image that can be safely moved and executed on top of different virtual machines.
- **Programming-level virtualization: I**mplemented by JVM or .NET runtime, binary code can be run without any recompilation.
- This makes flexible and very straight forward: One version is able to run on different platforms with no changes.
- Allows your own system always with you and ready to use as long as the required virtual machine manager is available (services you need available to you anywhere you go).

Characteristics of Virtualized Environments

1. Increased Security

✔ Controlled Execution for Guest Systems

- Virtualization provides a **secure and controlled** environment for guests (virtual machines).
- Each VM operates **independently** within an emulated environment.

✔ Guest-to-Host Security Filtering

- The **Virtual Machine Monitor (VMM)** ensures that guest operations are **translated and applied securely** to the host.
- It prevents harmful actions from affecting the host system.

✔ Isolation of Sensitive Information

- The host system remains **hidden from guests** by default.
- Sensitive data is protected without requiring complex security settings.

✔ Protection from Untrusted Code

- Applications downloaded from the Internet can be executed in a **sandboxed environment** (e.g., Java Virtual Machine - JVM).
- This limits their access to system resources, reducing security risks.

✔ Hardware Virtualization for Security

- Virtualization tools like **VMware, VirtualBox, and Parallels** create separate **virtual computers** on top of real hardware.
- The file system of a virtual machine is **fully separated** from the host system, adding an extra layer of security.

2. Managed Execution

✔ Sharing Resources Efficiently

- Virtualized data centers **share** resources between multiple virtual machines.
- Reduces the number of active physical servers, cutting **power consumption** and **hardware costs**.

✔ Aggregation of Hosts

- Multiple **physical hosts** can be grouped together and presented as a **single virtual host**.
- This technique is widely used in **cloud computing and distributed computing** for high efficiency.

✔ Emulation & Customization

- The virtualization layer **controls and fine-tunes** the environment that guests interact with.
- It allows different operating systems to run smoothly on shared hardware.

✔ Isolation for Stability & Security

- Multiple virtual machines **run independently** on the same host without interfering with each other.
- The **host system and guests remain separate**, ensuring **system stability and security**.

3. Portability (Easy Transfer & Compatibility)

✔ Hardware Virtualization for Mobility

- Virtual machines are packaged into **virtual images** that can be moved between different hardware setups.
- These images can run on any virtualized infrastructure **without modification**.

✔ **Programming-Level Virtualization**

- Technologies like **JVM (Java Virtual Machine) and .NET Runtime** allow software to run on different platforms **without recompilation**.
- This ensures software compatibility across multiple operating systems.

✔ **Cross-Platform Compatibility**

- A single software version can run on **multiple devices** without requiring changes.
- This makes software **flexible and future-proof**.

✔ **Accessibility Anywhere**

- With virtualization, your system and applications can be **accessed from anywhere**.
- As long as the required virtual machine manager is available, you can use your setup anywhere in the world.

🔹 Key Takeaway:

Virtualized environments **enhance security, optimize resource management, ensure portability, and improve system efficiency**, making them a crucial technology for **cloud computing, enterprise IT, and software development**.

Some extra info:

1. **What is a Virtual Machine Monitor (VMM)?**
    - The **VMM**, also known as a **hypervisor**, is a software layer that manages **virtual machines (VMs)** running on a physical system.
    - It ensures that multiple virtual machines **can run independently** on the same hardware.
2. **What does "Guest Operations" mean?**
    - **Guest** refers to the virtual machine (VM) running inside the system.
    - **Guest operations** include actions like running applications, reading/writing data, and interacting with hardware (CPU, memory, disk, network).
3. **What does "Translated and Applied Securely" mean?**
    - Since a **guest VM does not directly control hardware**, the **VMM translates** guest instructions into commands the physical hardware can understand.
    - The **VMM ensures security** by filtering out **harmful or unauthorized** actions, preventing the guest VM from harming the host system.

Example to Understand It Easily

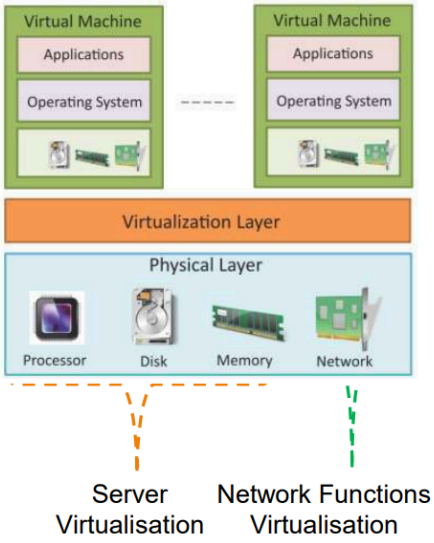
- Imagine a **VM running Windows** inside a **Linux host machine**.
- If the **Windows VM** wants to access the CPU, disk, or network, it sends a request to the **VMM** instead of directly accessing the hardware.
- The **VMM processes this request, checks if it's safe**, and then allows or denies the action.
- This prevents the **guest OS from interfering with the host OS**, ensuring system stability and security.

Key Takeaway

The **VMM acts as a security and control layer**, translating and managing virtual machine actions so that they do not interfere with or harm the **physical host system**. 🚀

Virtualization in Clouds

- Virtualization refers to the partitioning the resources of a physical computing hardware (such as computing, storage, network and memory) into multiple virtual resources or virtual machines (VMs).
- Key enabling technology of cloud computing that allow pooling of resources.
- In cloud computing, resources are pooled to serve multiple users using multi-tenancy.
- There are two major types of Virtualization approaches of cloud hardware at physical layer
  - Server Virtualization
  - Network Functions Virtualization



Explanation for above:

Virtualization in Clouds :

1. What is Virtualization in Cloud Computing?

- Virtualization **divides physical computing resources** (CPU, storage, memory, and network) into **multiple virtual resources**.
- These virtual resources are called **Virtual Machines (VMs)**, each running its own operating system and applications.

2. Why is Virtualization Important for Cloud Computing?

✔ Efficient Resource Utilization

- A single physical system can run **multiple VMs**, reducing hardware waste.

✔ Pooling of Resources

- Cloud computing **combines resources** (CPU, storage, memory) and shares them across multiple users.
- This technique is called **multi-tenancy**, where different users share the same physical infrastructure but have separate virtual environments.

✔ Flexibility & Scalability

- Cloud providers can quickly **increase or decrease virtual resources** based on user demand.

✔ **Cost Savings**

- Virtualization reduces the need for multiple physical servers, cutting costs on **hardware, maintenance, and power consumption**.

3. Two Major Types of Virtualization in Cloud Hardware

A. Server Virtualization

- **What is it?**
  - It creates **multiple virtual machines (VMs)** on a single physical server.
  - Each VM runs its own **operating system and applications**.
- **Benefits:**
  ✔ Efficient use of **CPU, memory, and storage**.
  ✔ Allows running **different operating systems (Windows, Linux) on the same hardware**.
  ✔ Ensures **isolation** between different VMs, improving security.

B. Network Functions Virtualization (NFV)

- **What is it?**
  - Instead of using **physical network devices** (routers, firewalls, load balancers), NFV virtualizes these functions.
  - Runs network services **as software on virtualized infrastructure**.
- **Benefits:**
  ✔ Reduces the need for **expensive physical networking equipment**.
  ✔ Allows **quick deployment of network services** in cloud environments.
  ✔ Improves **scalability** by adjusting network resources based on demand.

4. How Does Virtualization Work?

✔ **Virtualization Layer**

- Sits between the **physical hardware** and **virtual machines**.
- Manages the distribution of **CPU, memory, storage, and network resources** to VMs.

✔ **Physical Layer**

- Includes actual **hardware components** like **processors, disks, memory, and network**.
- These resources are divided into multiple virtual instances.

5. Real-World Use Cases

✔ Cloud Computing Services

- AWS, Google Cloud, and Microsoft Azure use virtualization to run multiple customers' workloads on shared hardware.

✔ Enterprise IT Infrastructure

- Companies use server virtualization to **reduce the number of physical servers** and improve efficiency.

✔ Telecommunication & Networking

- NFV helps telecom providers **deploy and manage network services faster** without needing physical routers or switches.

✔ Software Development & Testing

- Developers create **virtual testing environments** without needing multiple physical machines.

🔹 Key Takeaway:

Virtualization is a **core technology in cloud computing** that enables **efficient resource utilization, cost savings, scalability, and security**. It supports **server virtualization (VMs) and network function virtualization (NFV)**, making modern cloud infrastructure more **flexible and powerful**.

Virtual Machine (VM)

- VM can be implemented by adding a software layer to a real machine to support desired architecture, can be applied to entire machine, lie at architected interfaces.
- **Architecture:** a formal specification to an interface in the system, including the logical behavior of the resources managed via the interface.
- Implementation describes the actual embodiment of an architecture.
- Abstraction levels correspond to implementation layers, having its own interface or architecture.
- **Guest** – system component interacts with Virtualization Layer.
- **Host** – original environment where guest runs.
- **Virtualization Layer** – recreate the same or different environment where guest will run.

Explanation:

Virtual Machine (VM)

1. What is a Virtual Machine (VM)?

- A **VM is a software-based computer** that runs on top of a real (physical) machine.
- It acts as a separate computer with its own **operating system (OS), applications, and resources**.
- VMs are created using **a software layer called the Virtualization Layer**.

2. Key Components of a Virtual Machine

✔ Architecture

- Defines how the **VM interacts with hardware and software**.
- It includes the **logical structure** of resources like CPU, memory, and storage.

✔ Implementation

- Refers to how the **VM is actually created and runs** on a physical system.
- Uses **hypervisors** like VMware, VirtualBox, and Hyper-V.

✔ Abstraction & Layers

- A VM is built using **multiple layers of software** that allow it to function separately from the physical machine.
- These layers provide a **separate environment for applications** to run.

3. Key Elements in a Virtualized System

✔ Guest (Virtual Machine)

- The **virtual system** that runs on the virtualization layer.
- It has its own **OS, software, and applications**.

✔ Host (Physical Machine)

- The **real hardware** where the virtual machine runs.
- The host provides **CPU, memory, and storage** to multiple virtual machines.

✔ Virtualization Layer

- The software that **creates and manages virtual machines**.
- It **isolates each VM** and allows multiple VMs to run on a single host.

4. How Virtual Machines Work?

1️⃣ **A physical machine runs virtualization software** (Hypervisor).
2️⃣ The hypervisor **creates and manages multiple VMs**.
3️⃣ Each VM acts like a **separate computer**, with its own OS and applications.
4️⃣ The virtualization layer ensures that each VM gets a **share of CPU, memory, storage, and network**.

5. Benefits of Virtual Machines

✔ Efficient Resource Utilization

- Multiple VMs can share **the same physical hardware**, reducing waste.

✔ Cost Savings

- Fewer physical machines are needed, saving on **hardware, maintenance, and energy**.

✔ Security & Isolation

- Each VM is **independent**, so a failure or security breach in one VM **does not affect others**.

✔ Flexibility & Scalability

- New VMs can be **created, cloned, or deleted** as needed.

- Supports **different OS on the same machine** (e.g., running Windows and Linux together).

✔ **Easy Backup & Migration**

- A VM can be **saved, moved, or restored** easily, improving disaster recovery.

6. Real-World Uses of Virtual Machines

✔ **Cloud Computing**

- Cloud providers like **AWS, Azure, and Google Cloud** use VMs to host multiple users on shared hardware.

✔ **Software Development & Testing**

- Developers test applications on **different OS without needing separate machines**.

✔ **Enterprise IT Infrastructure**

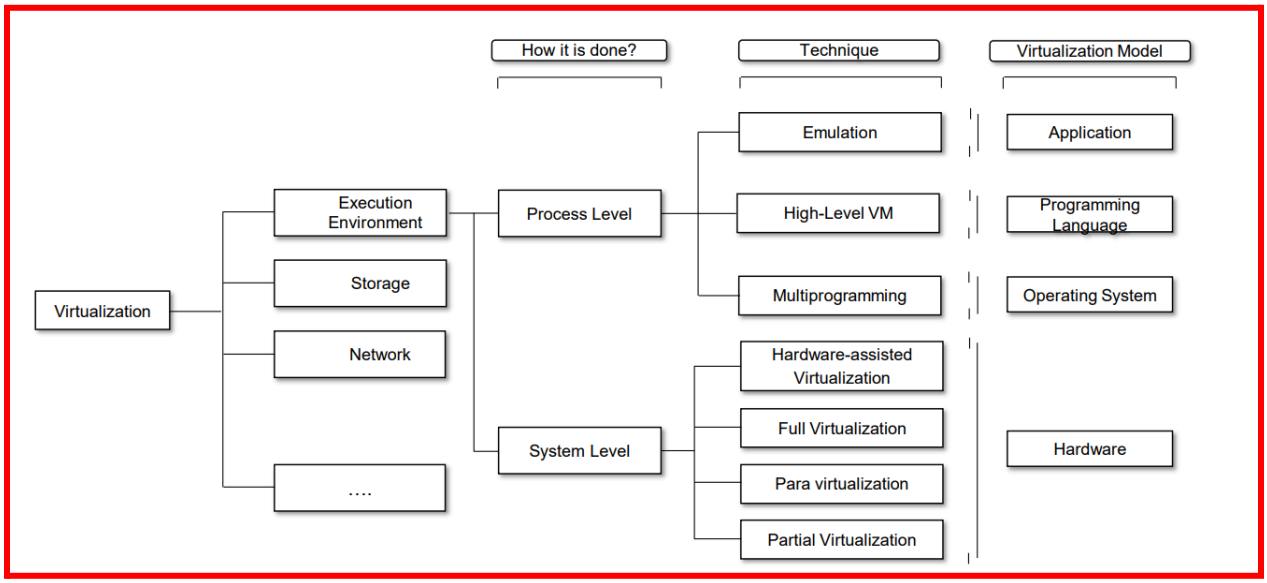
- Businesses use VMs to **run multiple workloads on fewer physical servers**.

✔ **Cybersecurity & Sandboxing**

- Security teams run suspicious programs inside VMs to **analyze malware safely**.

🔹 Key Takeaway:

A Virtual Machine (VM) **emulates a real computer inside a physical machine**, using a **virtualization layer** to create **separate, isolated environments**. VMs **improve resource efficiency, security, and flexibility**, making them essential in **cloud computing, software testing, and enterprise IT**.



Virtualization :::

1. What is Virtualization?

- Virtualization is the process of **creating virtual versions of resources** like computing environments, storage, or networks.
- It allows multiple operating systems or applications to **run on the same physical hardware**.

2. How Virtualization is Done?

Virtualization is applied in different areas:

✔ Execution Environment Virtualization

- A virtualized system where applications or processes run separately from the underlying hardware.

✔ Storage Virtualization

- Combines multiple **physical storage devices** into a single **virtual storage unit**.
- Example: Cloud storage systems (Google Drive, OneDrive).

✔ Network Virtualization

- Virtualizing network functions such as **firewalls, routers, and switches** to improve efficiency.
- Example: Software-defined networking (SDN), Virtual Private Networks (VPNs).

✔ **Other Virtualization Types**

- Includes virtualization of **memory, input/output (I/O), and security layers**.

3. Virtualization Techniques

Virtualization can be done using different techniques:

✔ **Process Level Virtualization**

- Virtualization at the application level where individual processes run in **isolated environments**.

✔ **System Level Virtualization**

- Creates a full **virtual machine (VM)** that runs an entire operating system separately.

✔ **Emulation**

- A system mimics another system's hardware or software environment.
- Example: Running **old video game consoles on modern computers**.

✔ **High-Level Virtual Machine (VM)**

- A complete system running inside another system.
- Example: Running **Windows OS inside a Mac using VirtualBox**.

✔ **Multiprogramming**

- A system where multiple processes share **CPU and memory resources** efficiently.
- Example: Running **multiple apps on a single OS simultaneously**.

✔ **Hardware-Assisted Virtualization**

- Uses **CPU support (Intel VT-x, AMD-V)** to improve virtualization performance.

  ✔ **Full Virtualization**

  - A **complete virtual machine** is created, fully independent of the host system.
  - Example: Running **Linux inside a Windows PC**.

✔ **Para-Virtualization**

- The guest OS **is aware it is running in a virtualized environment** and optimizes performance.
- Example: Xen Hypervisor.

✔ **Partial Virtualization**

- Only **some hardware components are virtualized**, while others run directly on the host.
- Example: VMware Workstation.

4. Virtualization Models

There are different types of virtualization models based on their purpose:

✔ **Application Virtualization**

- Allows running an application **without installing it on the local machine**.
- Example: Running **Google Docs in a web browser**.

✔ **Programming Language Virtualization**

- Allows programs to run across different platforms **without modification**.
- Example: Java Virtual Machine (JVM), .NET framework.

✔ **Operating System Virtualization**

- Allows multiple **OS to run on the same hardware**.
- Example: VMware, VirtualBox.

✔ **Hardware Virtualization**

- Creates a virtual version of **CPU, memory, disk, and network interfaces**.
- Example: Cloud computing platforms like **AWS, Azure, and Google Cloud**.

5. Key Benefits of Virtualization

✔ **Better Resource Utilization**

- Allows multiple applications and OS to share **the same hardware efficiently**.

✔ **Cost Savings**

- Reduces **hardware costs, power consumption, and maintenance**.

✔ **Security & Isolation**

- Each virtual machine is **separate**, preventing malware or crashes from affecting others.

✔ **Scalability & Flexibility**

- New virtual machines can be **created, modified, or deleted** easily.

✔ **Portability & Backup**

- Virtual Machines (VMs) can be **moved between systems** without reinstallation.

🔹 Key Takeaway:

Virtualization is a powerful technology that **creates virtual environments for applications, storage, networks, and hardware**, allowing **efficient resource management, security, and scalability** in cloud computing and IT infrastructure.

Hardware Virtualization

- It provides an abstract execution environment in terms of computer hardware on top of which a guest operating system can be run.
- The **guest** is represented by the operating system, the **host** by the physical computer hardware, the **virtual machine** by its emulation, and the **virtual machine manager** by the hypervisor (see Figure).
- The **hypervisor** is generally a **program** or a combination of **software** and **hardware** that allows the abstraction of the underlying physical hardware.
- Hardware-level virtualization is also called System level Virtualization.
- Since, it provides ISA to virtual machines, which is the representation of the hardware interface of a system.
- This is to differentiate it from process virtual machines, which expose ABI to virtual machines.

It provides an abstract execution environment in terms of computer hardware on top of which a guest operating system can be run.

- Virtualization allows a computer to create a "fake" or simulated environment that acts like real hardware. This allows multiple operating systems to run on the same physical computer.

The guest is represented by the operating system, the host by the physical computer hardware, the virtual machine by its emulation, and the virtual machine manager by the hypervisor (see Figure).

- "Guest" refers to the operating system running inside the virtual environment (like Windows running inside a virtual machine on Linux).
- "Host" is the actual, physical computer that provides resources like CPU and memory.
- "Virtual machine" is a simulated computer that behaves like a real one but runs inside the host system.
- "Virtual machine manager" (or hypervisor) is the software that manages and runs virtual machines.

The hypervisor is generally a program or a combination of software and hardware that allows the abstraction of the underlying physical hardware.

- A hypervisor is software (sometimes combined with specialized hardware support) that makes virtualization possible by managing virtual machines and ensuring they have access to resources like CPU, memory, and storage.

Hardware-level virtualization is also called System-level Virtualization.

- This type of virtualization happens at the hardware level, meaning the physical processor and memory are directly involved in supporting virtual machines. It is also called system-level virtualization.

Since it provides ISA to virtual machines, which is the representation of the hardware interface of a system.

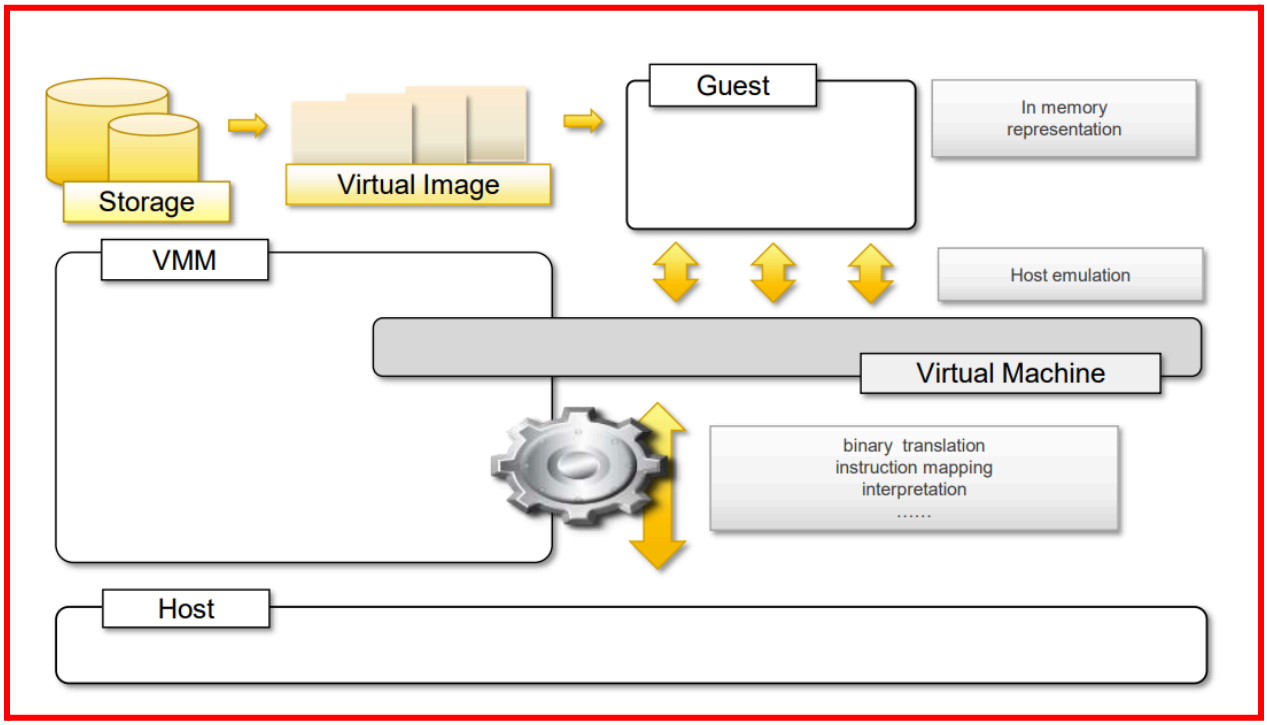
- ISA (Instruction Set Architecture) is the set of basic commands that a processor understands. In virtualization, the hypervisor provides a virtual version of ISA so that virtual machines think they are running on real hardware.

This is to differentiate it from process virtual machines, which expose ABI to virtual machines.

- There are different types of virtualization.
- System virtualization (what we are discussing here) emulates an entire computer.

- Process virtualization only creates an environment for a specific application to run, using an ABI (Application Binary Interface), which allows software to run on different platforms without modifying the code.

A hardware virtualization reference model



This diagram represents a **hardware virtualization reference model**

1. **Storage**
    - **The system stores virtual machine data in physical storage devices like hard drives or SSDs.**
2. **Virtual Image**
    - **A virtual image is a file that contains the entire operating system and application data, acting like a "blueprint" for a virtual machine. It is loaded from storage.**
3. **Guest**
    - **The guest represents the virtualized operating system running on the virtual machine (e.g., Windows running inside a VM on a Linux host).**
    - **It is loaded into memory as an active system.**

4. **In-Memory Representation**
    - **When the guest is running, it is stored in RAM (memory), just like any other operating system.**
5. **Host Emulation**
    - **The virtual machine manager (VMM) or hypervisor ensures that the guest OS can run on virtualized hardware by emulating the host system.**
6. **VMM (Virtual Machine Manager)**
    - **The hypervisor or Virtual Machine Manager (VMM) controls virtualization.**
    - **It acts as a bridge between the guest and the real hardware (host), ensuring they can communicate properly.**
7. **Virtual Machine (VM)**
    - **This is the simulated or emulated machine that runs the guest OS.**
    - **The VM behaves like a real computer, even though it is actually running on a shared physical system.**
8. **Binary Translation, Instruction Mapping, and Interpretation**
    - **The virtual machine uses techniques like binary translation and instruction mapping to ensure that software designed for physical hardware runs properly on a virtualized system.**
    - **These techniques allow software inside the VM to execute as if it were on real hardware.**
9. **Host**
    - **The actual physical machine that provides computing resources like CPU, RAM, and storage to support virtualization.**
    - **It runs the hypervisor to manage virtual machines.**

Flow of Virtualization Process:

1. **Data is stored in Storage as a Virtual Image.**
2. **The Virtual Image is loaded into memory, creating a Guest OS.**
3. **The Guest OS is an in-memory system managed by the VMM (Hypervisor).**
4. **The VMM ensures the guest OS runs properly by using host emulation.**
5. **The Virtual Machine (VM) is created, and it uses binary translation & instruction mapping to execute software.**
6. **The Host provides actual hardware resources to support this virtualized environment.**

This setup allows multiple virtual machines to run on a single physical machine, improving efficiency and resource utilization.

Hypervisors(or VMM):

- Hypervisor runs above the physical hardware.
- It runs in supervisor mode.
- It recreates a h/w environment in which guest operating systems are installed.
- It is a piece of s/w that enables us to run one or more VMs on a physical server(host).
- Two major types of hypervisor
- – ***Type -I***
- – ***Type-II***

Hypervisors (or VMM - Virtual Machine Manager):

1. **Hypervisor runs above the physical hardware.**
    - A hypervisor is software that operates on a physical computer (hardware) and enables virtualization.
2. **It runs in supervisor mode.**
    - The hypervisor has full control over the system's hardware, like the CPU and memory.
3. **It recreates a hardware environment in which guest operating systems are installed.**
    - The hypervisor creates a "fake" computer environment, so multiple operating systems can run simultaneously.
4. **It is a piece of software that enables us to run one or more VMs on a physical server (host).**
    - A hypervisor allows us to create and run multiple virtual machines (VMs) on a single physical computer.
5. **Two major types of hypervisors:**
    - There are two kinds of hypervisors that work differently.
    - **Type-I Hypervisor**
    - **Type-II Hypervisor**

Hypervisors (Type-I):

- It runs directly on top of the hardware, takes place of OS.
- Directly interact with the ISA exposed by the underlying hardware, and emulate this interface in order to allow the management of guest OS.
- Also known as *native virtual machine.*

Hypervisors (Type-2):

- It requires the support of an operating system to provide virtualization services.
- Programs managed by the OS, interact with OS through the ABI.
- Emulate the ISA of virtual h/w for guest OS.
- Also called hosted virtual machine.

No modification to Operating System code for both Type-1 and Type-2 hypervisors

Hypervisors (Type-I):

1. **It runs directly on top of the hardware, takes place of OS.**
    - This hypervisor is installed directly onto a computer and does not require an operating system. It acts as the system itself.
2. **Directly interacts with the ISA exposed by the underlying hardware, and emulates this interface in order to allow the management of guest OS.**
    - It directly communicates with the physical computer's CPU and hardware (Instruction Set Architecture - ISA) to create virtual machines.
3. **Also known as a native virtual machine.**
    - Because it works directly on hardware, without needing another OS, it is called a native hypervisor.

Hypervisors (Type-II):

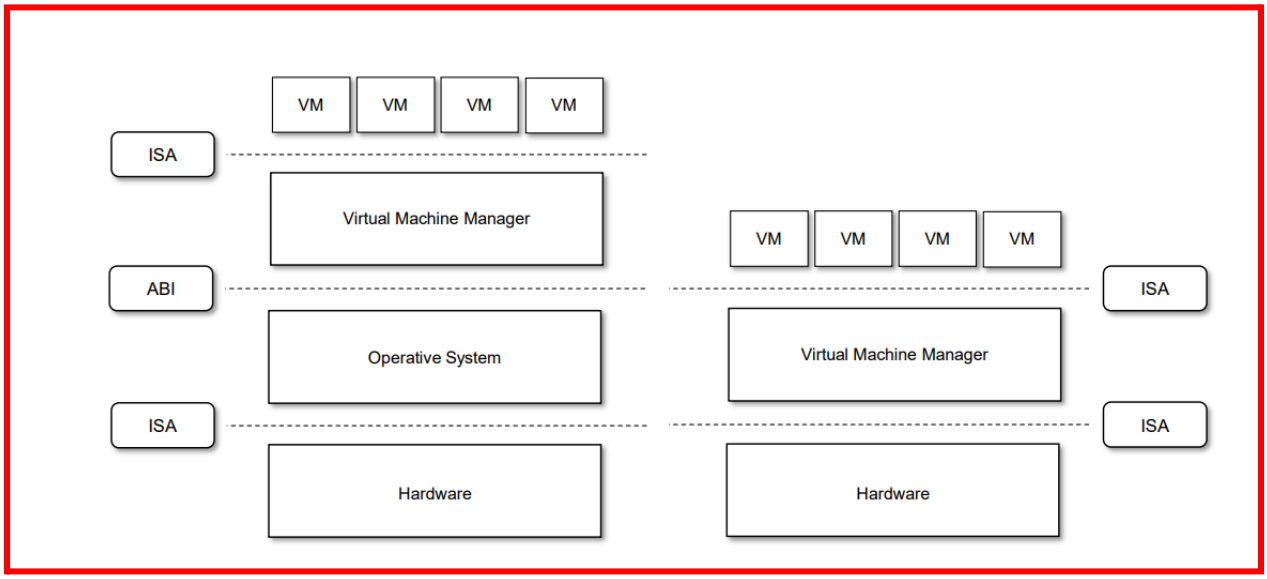
1. **It requires the support of an operating system to provide virtualization services.**
    - This type of hypervisor is installed **inside** an existing operating system (like Windows, Linux, or macOS) and then creates virtual machines.
2. **Programs managed by the OS, interact with OS through the ABI.**
    - The virtual machine runs like a regular application and communicates with the operating system through the ABI (Application Binary Interface).
3. **Emulate the ISA of virtual hardware for the guest OS.**

    - The Type-II hypervisor creates a **fake** hardware environment so that the virtual machines think they are running on real computers.
4. **Also called a hosted virtual machine.**
    - Since it runs on an existing operating system, it is called a "hosted" hypervisor.
5. **No modification to Operating System code for both Type-1 and Type-2 hypervisors.**
    - You don't need to change the OS to use either Type-I or Type-II hypervisors.

Key Differences:

- **Type-I Hypervisor**: Runs **directly on the hardware** and does not need an OS (e.g., VMware ESXi, Microsoft Hyper-V).
- **Type-II Hypervisor**: Runs **inside an operating system** like an application (e.g., VMware Workstation, VirtualBox).

Hosted and Native Hypervisors



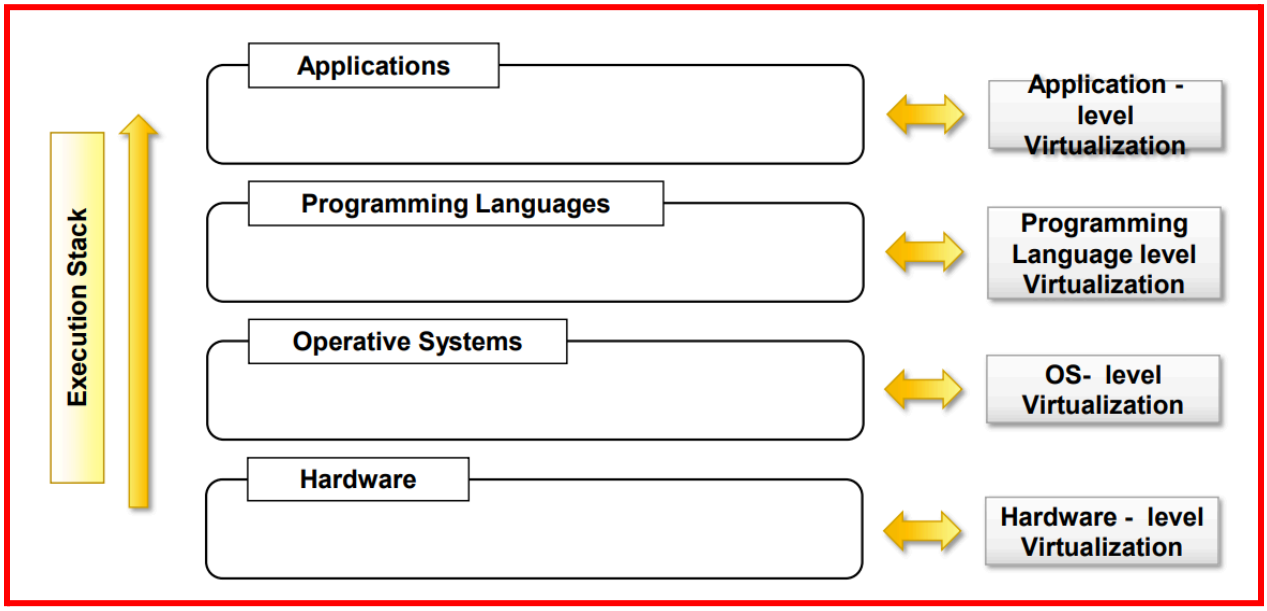
Execution Virtualization

- It emulates an Execution Environment (EE), separated from host, implemented on top of H/W by OS, application (libraries) dynamically or statically.

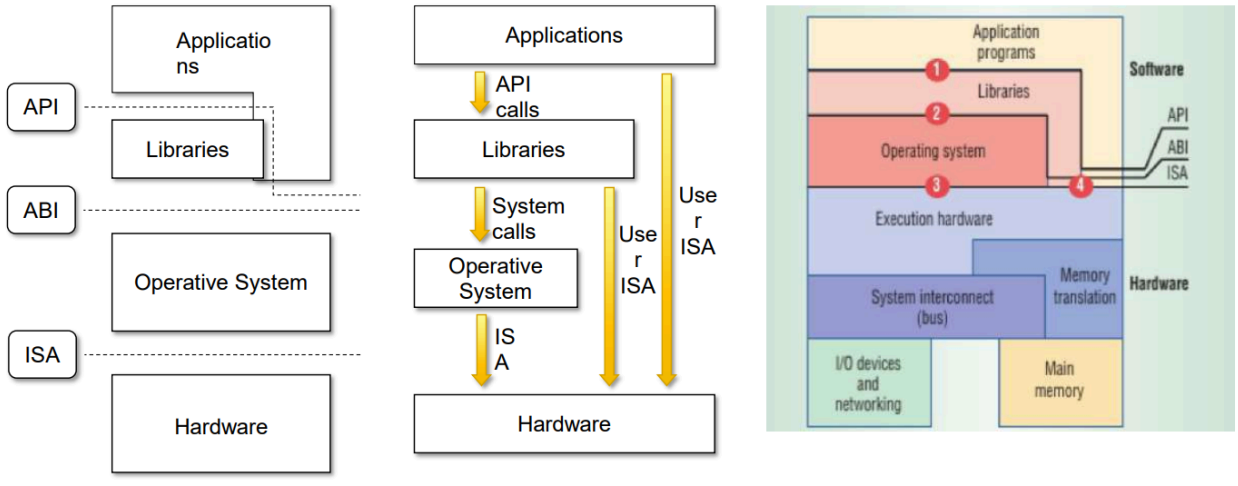
Machine reference model

- Virtualizing an execution environment at different levels of **computing stack** requires a reference model that defines interfaces between levels of abstractions.
- Virtualization techniques actually replace one of the layers.
- A clear separation between layers simplifies their implementation.
- It requires the emulation of interfaces and interaction with underlying layer.

Computing Stack



Machine Reference Model



- Depending on what is replaced /mimicked, we obtain different techniques of virtualization.

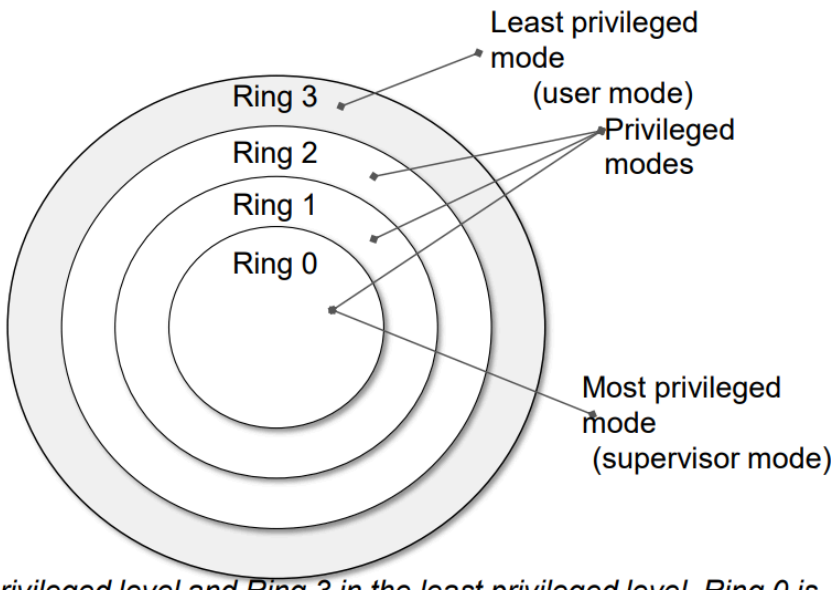
Execution virtualization: Interfaces

- **Virtualization:** extend or replace an existing interface to mimic the behavior of another system.
- **Instruction Set Architecture (ISA) :** processor, registers, memory and the interrupt management or H/W.
- **Application Binary Interface (ABI):** separates OS layer from application and libraries which are managed by the OS, System Calls defined, allows portability of applications and libraries across OS.
- **Application Programming Interfaces (API):** interfaces applications to libraries and/or the underlying OS.
- ISA has been divided into two security classes**: Privileged** Instructions and **Non-privileged** Instructions.

Execution virtualization

- **Non-privileged** instructions can be used without interfering with other tasks because they do not access shared resources (floating, fixed-point, and arithmetic instructions).
- **Privileged instructions** are executed under specific restrictions and mostly used for sensitive operations: behavior-sensitive: operate on the I/O) or control-sensitive: alter the state of the CPU.
- Some architectures feature more than one class of privileged instructions
- For instance, a hierarchy of privileges see (*Fig. Next Slide*), in the form of ring-based security: Ring 0, Ring 1, Ring 2, and Ring 3.

Security Rings and Privilege Modes



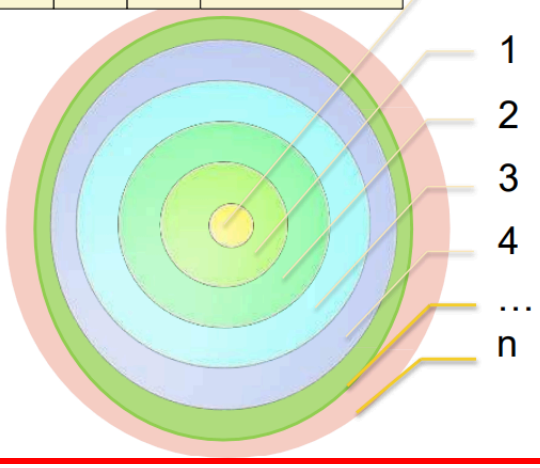
Ring 0 is in the most privileged level and Ring 3 in the least privileged level. Ring 0 is used by the kernel of the OS, rings 1 and 2 are used by the OS-level services, and Ring 3 is used by the user.

Security Rings and Privilege Modes

Segment Table revisited

… …	…		…
Start address	size	**ring**	Access type
…	…	…	…

= {read, write, execute}



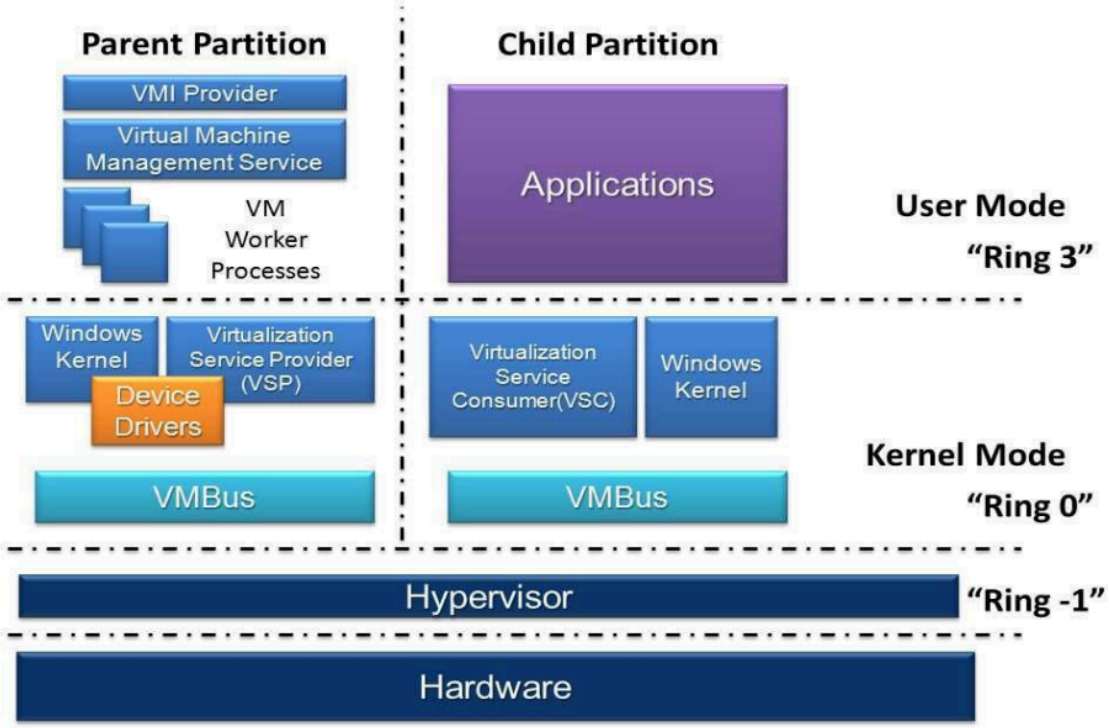
Code permitted to access ring *i* may access ring *j* if and only if *j* > *i*,

Execution virtualization

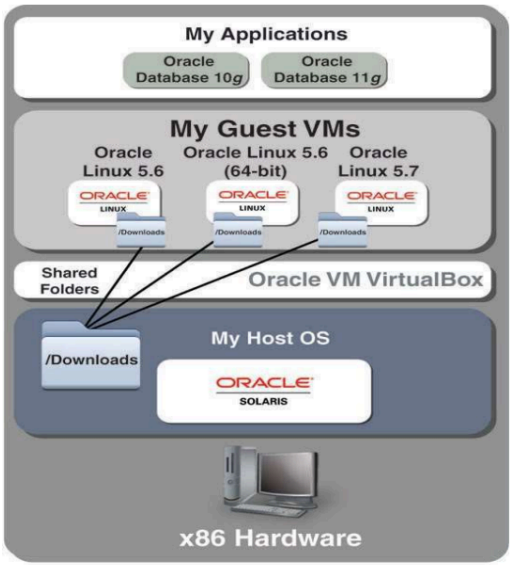
- Distinction between user & supervisor mode signifies role of the **hypervisor.**
- Hypervisors run in supervisor mode, and division between privileged and non-privileged instructions makes challenging design of virtual machine managers.
- Sensitive instructions execute in privileged mode, requires supervisor mode to avoid traps.
- Without this assumption it is impossible to fully emulate and manage the status of the CPU for guest operating systems.
- This is not true for the original ISA, allows 17 sensitive instructions to be called in user mode, prevents multiple OSs managed by a single hypervisor to be isolated from each other.
- Since, they access the privileged state of the processor and change it.

- Recent implementations of ISA (**Intel VT and AMD Pacifica**) solved this problem by **redesigning such instructions as privileged ones**.
- In hypervisor-managed environment, all guest OS code will run in user mode in order to prevent it from directly accessing the status of the CPU.
- If there are sensitive instructions that can be called in user mode, it is no longer possible to completely isolate the guest OS.

Type 1 Hypervisor CASE STUDY MICROSOFT HYPER-V



Type 2 Hypervisor



- VMWare example
  - Type 2 hypervisor Upon loading program: scans code for basic blocks
  - If sensitive instructions, replace by VMware procedure
    - Binary translation
  - Cache modified basic block in VMWare cache
    - Execute; load next basic block etc.
- Type 2 hypervisors work without VT (no automatic trapping of privileged instructions by hypervisors) support.

Hardware Virtualization

VMM: Main Modules coordinate to emulate the underlying hardware:-

Dispatcher: Entry Point of VMM

- Reroutes the instructions issued by VM instance.

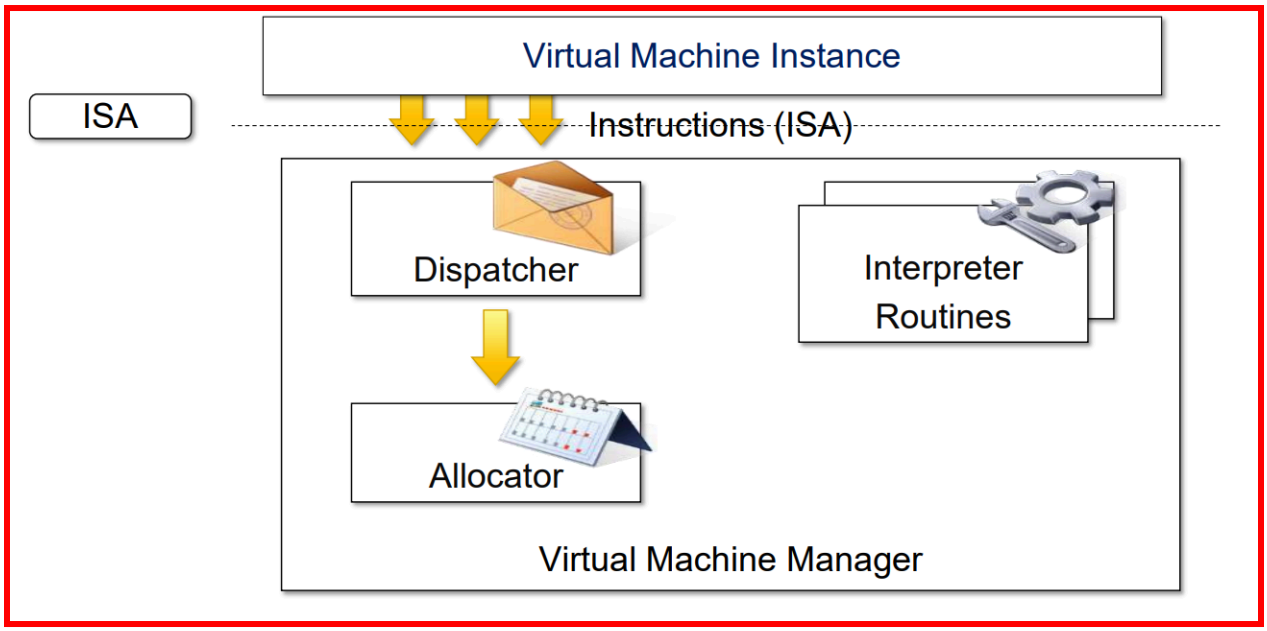
Allocator: Decides system resources to be provided to the VM;

- a VM executes an instruction, results in changing machine resources associated with that VM.
- Invoked by dispatcher

Interpreter: Consists of interpreter routines

- Executed whenever a VM executes a privileged instruction.
- Trap is triggered and the corresponding routine is executed.

VMM



Hardware Virtualization

Criteria of VMM*:

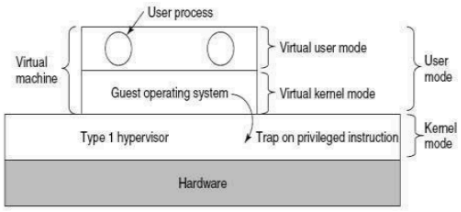
- **Equivalence** – same behavior as when it is ***executed directly*** on the physical host when it was running under control of VMM.
- **Resource control** – VMM should be in ***complete control of virtualized resources***.
- **Efficiency** – a statistically dominant fraction of the machine instructions should be ***executed without intervention*** from the VMM.
- **Theorems*: *Classification of the instruction set*** and proposed three theorems that define the properties that ***hardware instructions need to satisfy*** in order to efficiently support virtualization.
- Classification of IS:
- Privileged Instructions: trap if the processor is in user mode
- Control sensitive Instructions
- Behavior sensitive Instructions

* *criteria's are established by Goldberg and Popek in 1974*

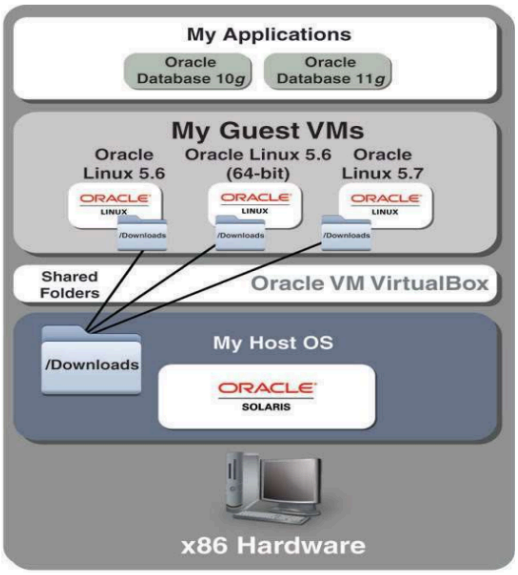
Type 1 Hypervisor

- Unmodified Operating System is running in user mode
  - But it thinks it is running in kernel mode (*virtual kernel mode)*
  - privileged instructions trap;
  - Hypervisor is the "real kernel"
    - Upon trap, executes privileged operations
    - Or emulates what the hardware would do

Ex: VMware ESXi or Microsoft Hyper-V



Type 2 Hypervisor



- VMWare example
  - Type 2 hypervisor Upon loading program: scans code for basic blocks
  - If sensitive instructions, replace by VMware procedure
    - Binary translation
  - Cache modified basic block in VMWare cache
    - Execute; load next basic block etc.
- Type 2 hypervisors work without VT (no automatic trapping of privileged instructions by hypervisors) support.

Hardware Virtualization

VMM: Main Modules coordinate to emulate the underlying hardware:-

Dispatcher: Entry Point of VMM

- Reroutes the instructions issued by VM instance.

Allocator: Decides system resources to be provided to the VM;

- a VM executes an instruction, results in changing machine resources associated with that VM.
- Invoked by dispatcher

Interpreter: Consists of interpreter routines

- Executed whenever a VM executes a privileged instruction.
- Trap is triggered and the corresponding routine is executed.

Hardware virtualization Techniques:

- CPU installed on the host is only one set, but each VM that runs on the host requires their own CPU.
- CPU needs to be virtualized, done by hypervisor.

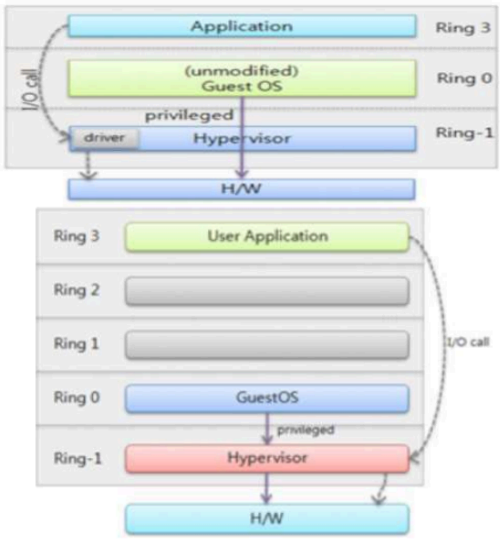
Types of HVT:

- **Full virtualization**
- **Hardware-assisted virtualization**
- **Para virtualization**
- **Partial virtualization**

Hardware Virtualization

Hardware-assisted virtualization:

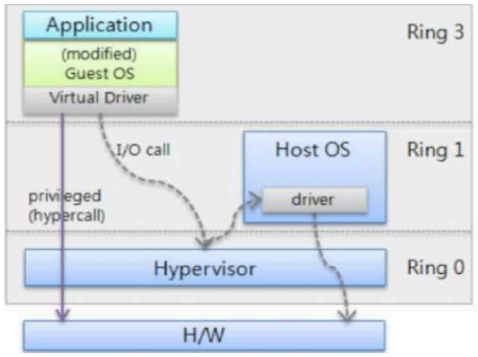
- In this hardware provides architectural support for building a VMM able to run a guest OS in complete isolation.
- ***Intel VT*** and ***AMD V*** extensions.
- Early products were using ***binary translation to trap some sensitive instructions*** and provide an emulated version.
- Additional Ring -1
- ***No binary translation*** of privileged instructions.
- Commands are executed directly to h/w via the hypervisor.



Para virtualisation

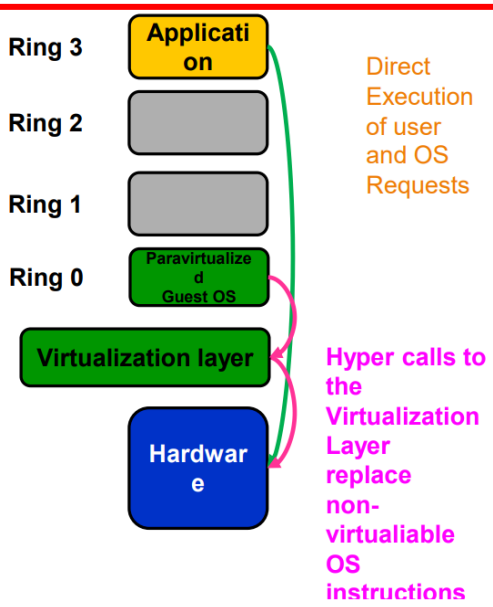
Para virtualization:

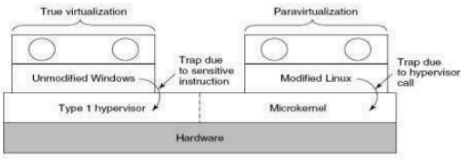
- Not-transparent virtualization
- – Thin VMM
- – Expose software interface to the virtual machine that is slightly modified from the host.
- – Guest OS needs to be modified.
- – Simply transfer the execution of instructions which were hard to virtualized, directly to the host.



- Privileged instructions of guest ***OS is delivered to the hypervisor*** by using hyper calls
- Hyper calls handles these instructions and accesses the h/w and return the result.
- Guest has authority to ***directly control*** of resources.

- Both type 1 and 2 hypervisors work on unmodified OS
- Para virtualisation: modify OS kernel to replace all sensitive instructions with hyper calls
  - OS behaves like a user program making system calls
  - Hypervisor executes the privileged operation invoked by hyper call.



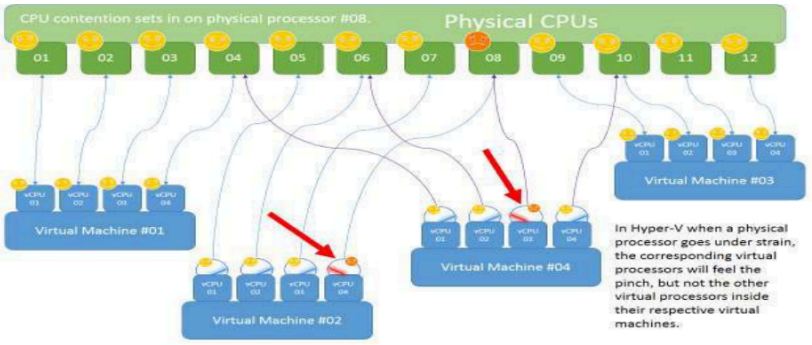


Partial virtualization:

- Partial emulation of the underlying hardware
- Not allow complete isolation to guest OS.
- Address space virtualization is a common feature of
- Contemporary operating systems.
- Address space virtualization used in time-sharing system.

Virtual CPU (VM) scheduling

- Based on fair-share balancing algorithm
  - Each VM gets equal time on the processor
- The VMs are allocated to physical processor core based on (Availability, Utilization, Priority (Research!!!))
- How privileged instructions requested by VMs depends on
  - Type of virtualization (Full virtualization, Para Virtualization, …)



Operating System-level virtualization

- To create different and ***separated execution environments*** for applications that are managed concurrently.
- No VMM or hypervisor.
- Virtualization is in single OS.
- OS kernel allows for multiple isolated user space instances.
- OS kernel is responsible for sharing the system resources among instances and other management.
- A user space instance has a proper view of the file system (isolated), separate IP addresses, software configurations, and access to devices.
- OS supporting virtualization are general-purpose, time-shared operating systems with the capability to provide resource isolation.

- Considered an evolution of the Chroot mechanism in Unix systems.
- Compared to H/W virtualization, no overhead because applications directly use OS system calls and there is no need for emulation.
- No need to modify applications, nor to modify any specific hardware.
- Good for server consolidation; multiple application servers share the same technology: operating system, application server framework etc.
- Different servers are aggregated into one physical server, each server is run in a different user space, completely isolated from the others. Ex: OpenVZ, IBM Logical Partition (LPAR), Solaris Zones and Containers.

Programming language-level virtualization

- To ease deployment of applications, managed execution, Security and portability across different platforms (OSs).
- A virtual machine executes the byte code of a program.
- VMs constitute a simplification of the underlying hardware instruction set and provide some high-level instructions that map some of the features of the languages compiled for them.
- At runtime, byte code can be either interpreted or compiled on the fly (JIT) against underlying hardware instruction set.
- Both Java and the Common Language Infrastructure (CLI), are stack-based virtual machines.
- Stack-based virtual machines possess the property of being easily interpreted and executed simply by lexical analysis and hence are easily portable over different architectures.

Application-level virtualization

- Applications run in runtime environments that do not natively support all the features required by such applications.
- Applications are not installed in the expected runtime environment but are run as though they were.
- Mostly concerned with partial file systems, libraries, and operating system component emulation.
- Such emulation is performed by a thin layer—a program or an operating system component—that is in charge
- Emulation can also be used to execute program binaries compiled for different hardware architectures.

Emulation strategies implemented:

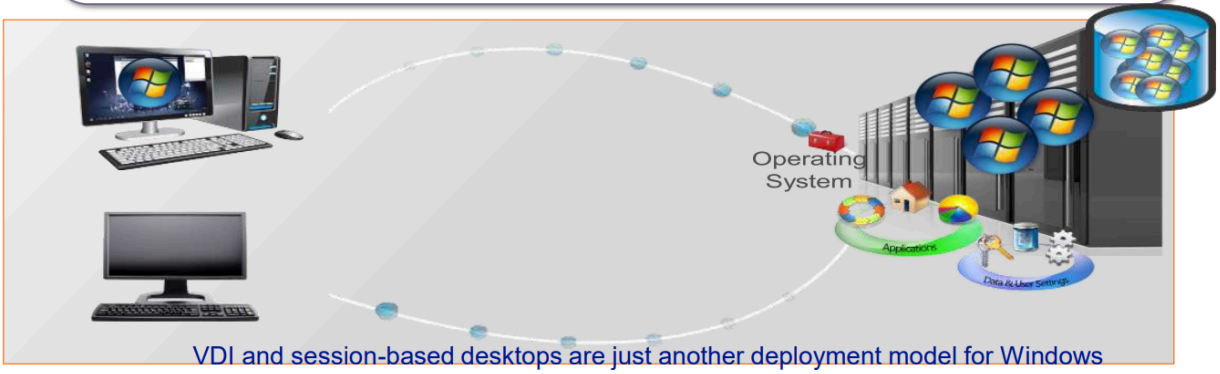
- **Interpretation**: source instruction is ***interpreted*** by an emulator for executing ***native ISA instructions***, ***minimal*** start up cost but ***huge overhead***.
- **Binary translation:** source instruction is ***converted to native*** instructions with equivalent functions.
- Block of instructions ***translated*** , ***cached*** and ***reused***.
- Large ***overhead cost*** , but over time it is subject to ***better performance***.
- In h/w virtualization , it allows the execution of a program ***compiled against a different h/w***.
- In Application level emulation , complete ***h/w environment***.

Other types of virtualization

- **Desktop virtualization** : abstracts the desktop environment available on a personal computer to access it using a client/server approach.
- Same outcome of hardware virtualization but serves a different purpose, makes accessible a different system as though it were natively installed on the host, but this system is remotely stored on a different host and accessed through a network connection.
- Same desktop environment accessible from everywhere.
- A highly available data center ensures the accessibility and persistence of the data, that is required.

What is a Virtual Desktop?

Virtual Desktop Infrastructure (VDI) and Remote Desktop Services session-based desktops are the key technologies that enable virtual desktops, whereby a desktop that runs in the data center can be delivered to the end-user's device using Remote Desktop Protocol (RDP). When combined with technologies that enable application and user state virtualization, organizations can achieve a high degree of desktop optimization and security and reduced TCO.



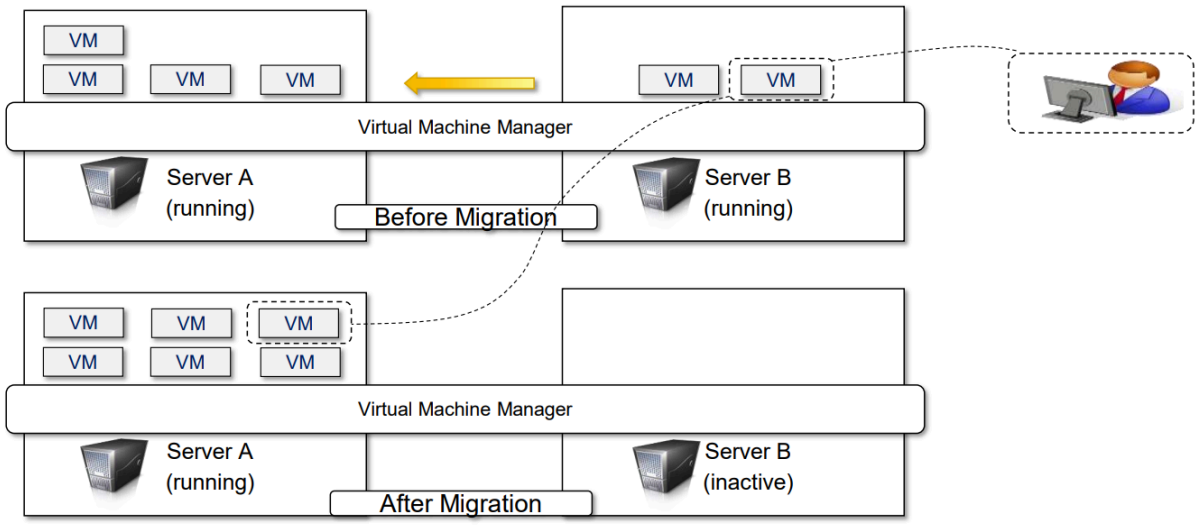
VDI and session-based desktops are just another deployment model for Windows

Other types of virtualization

- **Storage virtualization**: A system administration practices decoupling the physical organization of the hardware from its logical representation.
- No worried about the specific location of their data, it can be identified using a logical path.
- Harness a wide range of storage facilities and represent them under a single logical file system.
- Storage Area Network (SANs) use a network-accessible device through a large bandwidth connection to provide storage facilities.

- **Network Virtualization**: combines hardware appliances and specific software for the creation and management of a virtual network.
- Aggregate different physical networks into a single logical network (external network virtualization).
- Provide network-like functionality to an operating system partition (internal network virtualization)
- The result of external network virtualization is generally a virtual LAN.
- A VLAN is an aggregation of hosts that communicate with each other as though they were located under the same broadcasting domain.
- Internal network virtualization is applied together with hardware and operating system-level virtualization, in which the guests obtain a virtual network interface to communicate with.

Virtualization and Cloud Computing (VM Migration)



- Degree of **customization, security, isolation, and manageability.**
- Virtual computing environment
- Hardware virtualization is Infrastructure-as-a-Service (IaaS).
- Programming language virtualization is Platform-as-a-Service (PaaS) offerings.
- Attractive business opportunity for companies featuring a large computing infrastructure.
- Isolated and controllable environments
- **Server consolidation:** underutilized active resources reduced by aggregating VMs to small number of resources, can be fully utilized**.**
- Movement of virtual machine instances (VM **migration).**

MIGRATION:

1)“Live Migration of Virtual Machines”, Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, Andrew Warfield
2)“Post-Copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning”, Michael R. Hines and Kartik Gopalan

VM Migration

Review: Virtual Machines

Virtualization provides **interface identical to underlying bare hardware for each VM**
- i.e., all devices, interrupts, memory, page tables etc.

Virtualization Software: VMWare, Hyper-V, Oracle virtual box, Xen etc.

- It allows clean **separation between hardware and software**.
- **Process level migration problems can be avoided** by migrating a virtual machine.

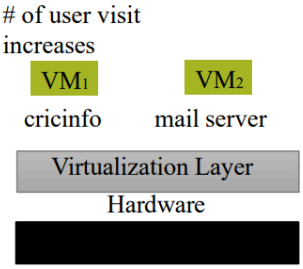
- **Virtualization provides facility to migrate** virtual machine from one host (source) to another physical host (destination).
- Virtual Machine Migration is a **useful tool for administrator** of data center and clusters.

Review: Virtual Machines in Cloud

- **Benefits** of Virtual Machines
  - Virtualization helps in making **efficient use of hardware resources**
  - Facilitates a greater **degree of abstraction**
  - **Easily move** from one piece of hardware to another
  - **Replicate** them at will
  - Create more **scalable and flexible infrastructure**
  - **High Availability and Fault tolerance (How?)**
- Cloud computing has taken that **degree of efficiency and agility realized** from virtualization
  - Pooled resources
  - Geographic diversity
  - Universal connectivity

Motivation

- Consider a **data center** consisting of **"one" physical machines (PM) hosting "two" VMs implementing one customer application each**
- **Resources**(CPU, Network, Memory, I/O) are **allocated to each VM** to **handle the workload** and operate at certain performance level **(SLA)**
- Each **VM sees workload fluctuation (WHY?)** from time to time => resource requirement changes



PM Capacity:
- Network Bandwidth = 10 Gbps
- Memory = 16 GB
- CPU = 8 cores

Resource Allocation:

	VM1	VM2
	N = 5Gbps	N = 5Gbps
	M = 8GB	M = 8GB
	C = 4 cores	C = 4 cores

Motivation

- An **increase in workload** can be handled by **allocating more resources** to it, **if idle** resources are available

❑ Main Issues:

What if PM does not have (enough or no) idle resources to satisfy VM's requirement?

- **Performance** of the application degrades
- **SLA violation** occurs

❑ **Key Ideas**

- Replication VMs
- Migrating VMs

Virtual Machine Migration

- Why do we need migration?
- When do we need to migrate?
- How migration is done?
- Issues in long distance migration (across data centers)

Why do we need migration?

- To Maintain/Upgrade
- To Optimize the Performance
- To Protect SLA
- To Improve QoE

When we need to migrate?

- **Hotspots can cause SLA violations**
  - Burden on some Virtual or Physical Machines are called hotspots

- **Hotspot Detection**
  - ***Black-box Monitoring***
    - CPU
    - Network
    - Memory
  - ***Gray-box Monitoring***
    - Gather OS level statistics and application logs

- A hotspot is **flagged only if thresholds or SLAs are exceeded** for a sustained time

- **SLA violation detection**
  - **Mapping** low-level resource metrics to high-level SLAs
    - ***CPU speed maps to Response Time***
    - ***Occupied memory size maps to number of concurrent clients***
  - **Predictive Strategy** for detection of possible SLA violations
  - **Detection interval**
    - **Short measurement intervals** may **degrade** performance
    - **Long measurement intervals** may cause **ignorance of heavy SLA** violations.

Migration Techniques

- The **different virtual machine migration** techniques are as follows:
  - Fault Tolerant Migration Techniques
  - Load Balancing Migration Techniques
  - Energy Efficient Migration Techniques

V M Migration Methods

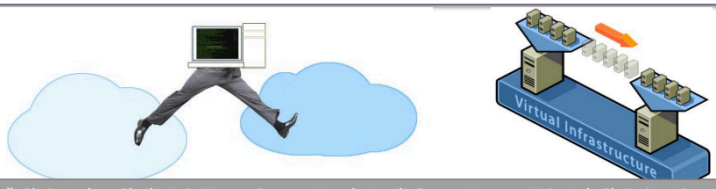
- Virtual Machine Migration methods are divided into **two types:**
  - **Cold (non-live) migration**
  - **Hot (live) migration**
- The **status of the VM loses** and **user can notice the service interruption** in cold migration. First, **VM is suspended**, then its **state is transferred**, and lastly, **VM is resumed at destination** host.
- In **live migration process, the state of a running VM is transferred/migrated from Host A to Host B.**
- **VM keeps running** while migrating and **does not lose its status. User doesn't feel any interruption** in service in hot (live) migration.

Live (Hot)VM Migration

- VM live migration can be a extremely **powerful tool for cluster/system administrators.**
  - Hardware / Software maintenance / upgrades
  - Load balancing / resource management
  - Distributed power management



- **Move VM instances across distinct physical hosts** with **little or no downtime** for running services.
  - **Services are unaware** of the migration.
  - Maintain **network connections of the guest OS**.
  - **VM is** treaded as a **black box**.
  - **VM (OS level) Migration is easier than migrating processes**



"Live Migration of Virtual Machines", Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, Andrew Warfield
"Post-Copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning", Michael R. Hines and Kartik Gopalan

Why OS-level migration, instead of process-level?

- ***Avoid 'residual dependencies'***
  - Original host can be power-off / sleep once migration completed.
- ***Can transfer in-memory state in a consistent and efficient fashion***
  - E.g., No reconnection for media streaming application
- ***Allow a separation of concerns b/w the users and operator of a cluster***
  - Users can fully control of the software and services within their VM.
  - Operators don't care about what's occurring within the VM.

What is migrated?

- **CPU context of VM, contents of main memory**
  - Narrow interface, easier than process migration
- **Disk: assume NAS (network attached storage) that is accessible from both hosts, or local disk is mirrored**
  - We do not consider migrating disk data
- **Network: assume both hosts on same LAN**
  - Migrate IP address, advertise new MAC address to IP mapping via ARP reply
  - Migrate MAC address, let switches learn new MAC location
  - Network packets redirected to new location (with transient losses)
- **I/O devices are provisioned at target**
  - Virtual I/O devices easier to migrate.

Design-local resources